

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

NÚCLEO DE EDUCAÇÃO A DISTÂNCIA

Pós-graduação *Lato Sensu* em Arquitetura de Software Distribuído

Felipe Jose Balbino / Gabriel da Silva Ferreira

SISTEMA DE GESTÃO AMBIENTAL

Rio de Janeiro

2020

Felipe Jose Balbino / Gabriel da Silva Ferreira

SISTEMA DE GESTÃO AMBIENTAL

Trabalho de Conclusão de Curso de Especialização em Arquitetura de Software Distribuído
como requisito parcial à obtenção do título de especialista.

Orientador: Luiz Alberto Ferreira Gomes

Tutora: Marjorie Pimentel

Rio de Janeiro

2020

AGRADECIMENTOS

Durante o nosso caminho até aqui foram muitas as pessoas que nos incentivaram, as quais devemos minha sincera gratidão.

Gostaria de agradecer primeiramente aos nossos pais, por sempre me incentivarem e me apoiarem nos estudos. Somos gratos também por toda a paciência que tiveram conosco em todos os momentos de desespero, impaciência e ausência. Somos eternamente gratos por sempre afirmarem que eu somos capazes de conquistar todos os meus sonhos e objetivos.

Reconheço que não há palavras para descrever o quanto somos gratos aos professores que fizeram parte da nossa formação. Sem o reconhecimento e incentivo de todos os professores que fizeram parte da nossa vida, nós não teríamos chegado até aqui.

Agradeço também aos amigos e familiares que nos deram a oportunidade de desenvolver nosso trabalho e aplicar nossos conhecimentos proporcionando-nos o amadurecimento profissional necessário. A todos os colaboradores pelo empenho em nos conceder as informações necessárias para alcançar com êxito nosso objetivo.

RESUMO

Depois da tragédia no rompimento de barragem em Brumadinho, criou-se uma nova maneira de encarar as questões ambientais, o foco foi alterado para o meio-ambiente e na prevenção de acidentes em instalações de mineradoras situadas em todo Brasil. Qualquer mau funcionamento dessas estruturas pode ocasionar uma anomalia grave ou até mesmo um rompimento, causando perdas econômicas, ambientais e a pior de todas: a perda de vidas humanas. Diante desta nova proposta de conscientização, a educação ambiental se tornou crucial para formação de indivíduos capazes de compreender o mundo e agir nele de forma consciente e responsável. Assim, este trabalho trata do levantamento dos principais ganhos ambientais quando existe um olhar diferenciado dentro de uma estrutura de negócio, tendo como base a melhoria contínua, visto que a preservação do meio ambiente, o uso racional de recursos naturais e a mudança de posturas da sociedade frente às questões ambientais têm levado as indústrias a buscar um melhor desempenho nessa área. Aliados a esses fatores, está à constatação de que a falta de uma gestão analítica, resultam em grandes impactos socioambientais. Sumariamente a solução proporcionará um desempenho de análise extremamente elevado assim mantendo os custos baixos, alto índice de assertividade e previsibilidade ao permitir confiança na tomada emergencial de decisão. O objetivo deste trabalho é demonstrar que existem ganhos, quando temos possibilidade de usar informações e dados da forma correta.

Palavras-chave: arquitetura de software, projeto de software, requisitos arquiteturais, design de software.

SUMÁRIO

1. Objetivos do trabalho	8
2. Descrição geral da solução	8
2.1. Apresentação do problema	8
2.2. Descrição geral do software (Escopo)	9
3. Definição conceitual da solução	10
3.1. Requisitos Funcionais	10
3.2 Requisitos Não-Funcionais	12
3.3. Restrições Arquiteturais	16
3.4. Mecanismos Arquiteturais	17
4. Modelagem e projeto arquitetural	19
4.1. Modelo de casos de uso	19
4.2. Modelo de componentes	24
4.3. Modelo de implantação	26
4.4. Modelo de dados	27
5. Prova de Conceito (POC) / protótipo arquitetural	30
5.1. Implementação e Implantação	30
5.2. Interfaces/ APIs	32
6. Avaliação da Arquitetura	40
6.2. Cenários	40
6.3. Avaliação	41
6.4. Resultado	53
7. Conclusão	55
REFERÊNCIAS	56
APÊNDICES	57

1. Objetivos do trabalho

O objetivo geral deste projeto arquitetural é oferecer uma proposta para uma plataforma de gestão ambiental com foco em uma implementação de um ecossistema de micro serviços disruptivos e modulares, com seu ambiente provendo disponibilidade para Websites e Mobile. Oferecer aos usuários uma interface diferenciada de sistema, realizando uma integração de todos os processos existentes, criando um controle maior para parte das empresas mineradoras e organizações que as fiscalizam, possibilitando uma maior integração entre o campo e os agentes controladores.

Os objetivos específicos são:

1. Criar o módulo de cadastro de ativos, para que os mesmos possam manter e modificar os seus insumos, máquinas e equipamentos. Solicitar alteração e inserção dos ativos e manter uma gestão automatizada dos equipamentos, a fim de efetuar agendamentos de manutenções preventivas e corretivas, para que sejam realizadas segundo o cronograma previamente inserido. Esse módulo será restrito para o perfil de administrador do sistema, e o controle dessas operações é acessado através de autenticação segura.
2. Criar um módulo de controle de processos minerários para realizar o cadastro dos fluxos de trabalho de diferentes setores dentro da demanda de exploração de minério, registrando os tipos de processos, além de suas paradas e problemas diários.
3. Criar um módulo de monitoramento de barragens, onde os usuários poderão cadastrar, editar e inativar sensores, tipos de sensores e barragens. Além da marcação das inspeções realizadas nas barragens pelos engenheiros.

2. Descrição geral da solução

2.1. Apresentação do problema

O principal objetivo das organizações empresariais é o lucro, e na busca ávida pelo máximo, muitas das vezes é negligenciado políticas e processos claros de restrições ambientais e manutenibilidade eficientes para segurança e requisitos básicos de uma organização sustentável. E esse processo fundamental ainda é arcaico, caro, ineficiente e

demorado, pois muitas das vezes, não utilizam sistema informatizados, ou não o fazem por falta de processos definidos na estrutura da corporação.

Nesse sentido, essas atividades dependem de sistemas com altos níveis de gerenciamento e maturidade para utilizar uma estrutura capaz de manter e direcionar altos volumes de informação em um período extremamente curto de tempo, pois os dados sofrem alterações constantes, que representam tomadas de decisões, que são o suporte essencial para segurança do negócio e do meio-ambiente.

Com uma análise mais eficiente da manutenibilidade dos ativos, seria possível manter um controle minucioso dos possíveis acidentes, a fim de amenizar determinados riscos inerentes ao processo de sustentação da barragem e seus equipamentos. Plano de ações e notificações podem ser utilizados para mitigar o risco de eventuais acidentes e evacuações.

2.2. Descrição geral do software (Escopo)

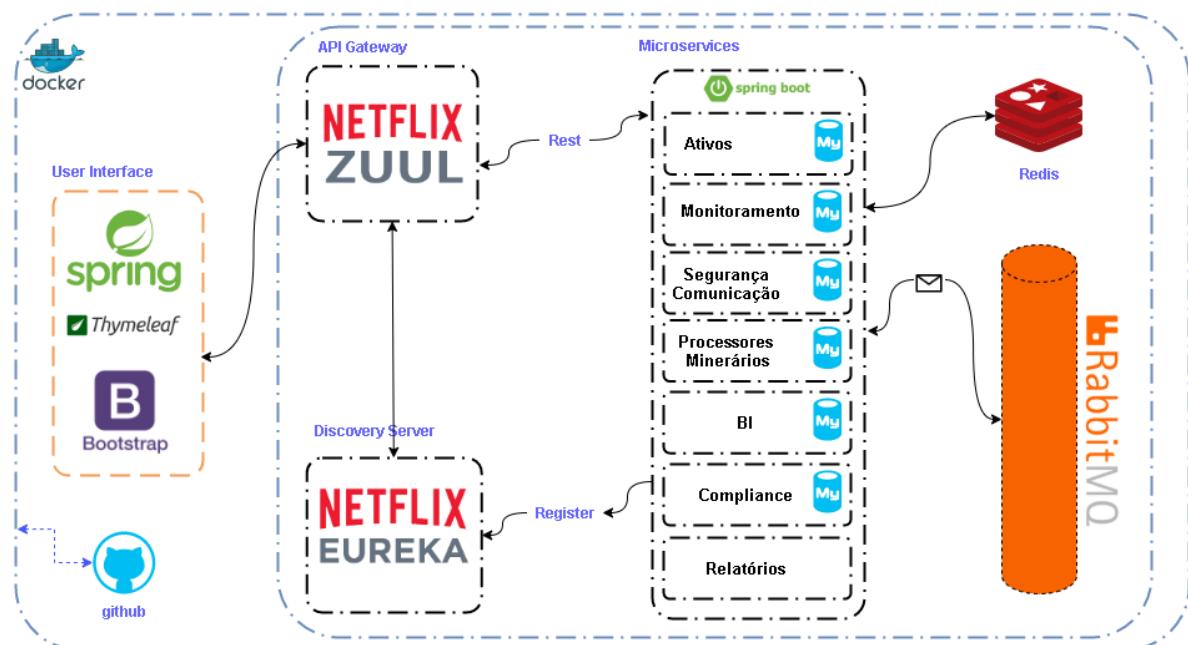


Figura 1: Diagrama informal da arquitetura

O software tem como objeto principal prover uma gestão e automação melhoradas a fim de efetivar um maior controle sobre o negócio e melhorar as atividades de gestão de mineradoras e automação do controle ambiental.

Como estrutura principal, temos uma interface do usuário autônoma, que se comunica diretamente aos serviços disponíveis, que são responsáveis pelo retorno de informações. A

aplicação possui arquitetura baseada em micro serviços, separados em diferentes módulos: Módulo para geração de relatórios; Módulo para cadastro dos ativos; Módulo para Controle de Monitoramento; Módulo de segurança e comunicação; Módulo de Notificação; Módulo de Gateway; no qual cada um possui uma função distinta e podendo existir de forma autônoma e auto gerenciada.

Como descrição geral do software, o sistema deverá gerenciar os riscos inerentes ao processo de mineração e ajuntamento de rejeitos, embasados por inspeções de Engenheiros utilizando os dados fornecidos por sensores e relatórios analíticos, emitindo alertas aos usuários que estão em áreas consideradas delicadas ao fluxo de funcionamento da mineradora. Dados são auditados para que os processos da mineradora estejam de regular perante as Normas Ambientais e o meio ambiente.

O sistema será acessado por 3 diferentes perfis:

- Administrador: Perfil com mais amplitude do sistema. Permite cadastrar, atualizar, visualizar e excluir qualquer que seja o processo em produção. Podendo emitir análises técnicas e tomadas de decisões específicas, auxiliando o negócio através de relatórios e dashboards.
- Engenheiro: Perfil responsável por monitorar o processo técnico, como registro de comunicações, conformidade de segurança e normas ambientais. Também é de responsabilidade do Engenheiro a geração de notificações técnicas para serem disponibilizadas publicamente.
- Consultor: Perfil responsável por acompanhar o processo relacionado os tipos de manutenções cadastradas aos ativos, e além de emitir alertas programados e reagir a desconformidades do processo de acompanhamento das barragens e sensores em produção.

3. Definição conceitual da solução

3.1. Requisitos Funcionais

- **Módulo de controle de usuários**
 - O sistema deve permitir o cadastro de novos usuários.
 - O sistema deve permitir que o usuário cadastrado realize o login.
 - O sistema deve permitir que o usuário realize o logout.

- o O sistema deve permitir o controle de acesso dos usuários com o objetivo de limitar o acesso a seções do sistema de acordo com suas responsabilidades.
- **Módulo de controle de ativos**
 - o O sistema deve permitir o cadastro de máquinas e equipamentos.
 - o O sistema deve permitir a gestão dos equipamentos e, com base em parâmetros pré-definidos, possibilitar o agendamento de manutenções preventivas e corretivas.
 - o O sistema deve possuir Integração com o sistema de aquisições para solicitação de serviços ou materiais de fornecedores externos.
 - o O sistema deve permitir o cadastro de insumos.
- **Módulo de controle de processos minerários**
 - o O sistema deve permitir o cadastro de fluxos de trabalho de acordo com o tipo de minério e método de lavra utilizado com o objetivo de entregar uma exploração eficiente e ambientalmente correta.
 - o O sistema deve permitir o cadastro de paradas e problemas ocorridos na produção.
- **Módulo de monitoramento de barragens**
 - o O sistema deve permitir o cadastro de barragens.
 - o O sistema deve permitir o cadastro de sensores e sua vinculação com as barragens.
 - o O sistema deve permitir o cadastro de tipos de incidentes.
 - o O sistema deve possuir integração com o sistema da Defesa Civil municipal e estadual.
 - o O sistema deve monitorar as medições recebidas pelos sensores e, comparando com dados previamente configurados, emitir um alerta de incidente ao módulo de segurança e comunicação.
- **Módulo de segurança e comunicação**
 - o O sistema deve permitir cadastrar contatos de pessoas que vivem próximo às barragens.
 - o O sistema deve emitir um alerta de segurança às pessoas e à Defesa Civil caso o módulo de monitoramento de barragens dispare um incidente onde seja necessário um procedimento de evacuação.
- **Módulo de inteligência do negócio (BI)**

- o O sistema deve processar os dados oriundos dos outros módulos consolidando o que for relevante para apresentar relatórios e dashboards que serão utilizados em futuras tomadas de decisões.
- **Módulo de compliance**
 - o O sistema deve permitir o acesso às informações de normas nacionais e internacionais do setor mineral, estas sendo integradas diretamente de sistemas externos evitando assim a defasagem de informação.
- **Módulo de relatórios de acompanhamento**
 - o O sistema deve permitir a geração de relatórios de incidentes.
 - o O sistema deve permitir a geração de relatórios de paradas e problemas ocorridas nos fluxos de trabalho dos processos minerários.

3.2 Requisitos Não-Funcionais

- **Acessibilidade:** O sistema deve possuir o layout responsivo, com a finalidade de ter o acesso possível via desktop ou dispositivos móveis.

Estímulo	Usuário acessa o sistema
Fonte de estímulo	Usuário acessando o sistema a partir do browser de um celular
Ambiente	Produção, carga normal
Artefato	Todos os módulos do sistema
Resposta	O layout do sistema se adapta a resolução do aparelho e mantém a usabilidade
Medida da resposta	Todos os módulos mantêm as mesmas funcionalidades, mesmo que adaptando o tamanho dos campos e elementos de tela para uma boa apresentação em resoluções diferentes

- **Usabilidade:** O sistema deve possuir boa usabilidade e interface intuitiva.

Estímulo	Usuário cobrastra um novo ativo
Fonte de estímulo	Usuário utilizando o módulo de ativos do sistema

Ambiente	Produção, carga normal
Artefato	Módulo de ativos
Resposta	O sistema possui uma interface intuitiva e padronizada. É possível realizar o cadastro facilmente com poucas interações do usuário.
Medida da resposta	Usuário consegue realizar o cadastro de um novo ativo em menos de cinco minutos

- **Desempenho:** O sistema deve possuir respostas rápidas as ações do usuário.

Estímulo	Usuário lista os ativos do sistema
Fonte de estímulo	Usuário listando os ativos cadastrados do sistema
Ambiente	Produção, carga normal
Artefato	Módulo de ativos
Resposta	O sistema apresentou todos os dados solicitados
Medida da resposta	A tempo de resposta do sistema foi inferior a 4 segundos

- **Disponibilidade:** O sistema deve possuir alta disponibilidade, 24 horas por dia e 7 dias por semana

Estímulo	Shutdown no cluster principal do servidor
Fonte de estímulo	Administrador do servidor de aplicação
Ambiente	Produção, carga normal
Artefato	Gerenciador de cluster
Resposta	Usuários que estavam logados no sistema continuam a utilizar sem haver nenhuma mudança
Medida da resposta	As solicitações ao cluster desligada são encaminhadas e processadas ao outros clusters disponíveis, sem grandes impactos na utilização

- **Manutenibilidade:** O sistema deve ser fácil de realizar correções em produção

Estímulo	Ajuste no módulo de ativos
Fonte de estímulo	A necessidade de inclusão de um novo campo no formulário de cadastro de ativos
Ambiente	Produção, carga normal
Artefato	Módulo de ativos
Resposta	A alteração do módulo é realizada sem impactos
Medida da resposta	São realizados os testes unitários da nova alteração, então é criado uma nova imagem Docker e realizado o deploy no ambiente de produção

- **Manutenibilidade:** O sistema deve ser fácil de ter suas funcionalidades testadas

Estímulo	Testes unitários
Fonte de estímulo	Integração contínua
Ambiente	Homologação
Artefato	Todos os módulos do sistema
Resposta	Execução de testes unitários de todos os módulos do sistema
Medida da resposta	Após a aprovação dos testes unitários o código é promovido ao ambiente de homologação para a validação das alterações.

- **Interoperabilidade:** O sistema deve se comunicar com sistemas externos.

Estímulo	Visualizar normas ambientais
Fonte de estímulo	Visualização das normas ambientais obtidas diretamente da API da Agência Nacional de Mineração
Ambiente	Produção, carga normal
Artefato	Módulo de compliance
Resposta	As informações foram visualizadas corretamente

Medida da resposta	O sistema acessou a API da Agência Nacional de Mineração e disponibilizou as informações na tela ao usuário
---------------------------	---

- **Interoperabilidade:** O sistema deve receber os dados dos sensores

Estímulo	Recebimento de dados dos sensores
Fonte de estímulo	Sensores de monitoramento
Ambiente	Produção, carga normal
Artefato	Módulo de monitoramento de barragens
Resposta	Recebe os dados dos sensores e, se necessário, chama o módulo de segurança e comunicação para a notificação de alertas
Medida da resposta	Comunicação entre os módulos para emissão de alertas

- **Interoperabilidade:** O sistema enviar email de notificação de alerta

Estímulo	Recebimento uma notificação de alerta
Fonte de estímulo	Módulo de monitoramento de barragens
Ambiente	Produção, carga normal
Artefato	Módulo de Segurança e Comunicação
Resposta	Recebe a notificação de uma alerta e envia um email de acordo com o que foi configurado no plano de ação
Medida da resposta	Envio de email correto

- **Segurança:** O sistema deve ter alta segurança de seus módulos, disponibilizando o acesso somente aos usuários autorizados.

Estímulo	Acessar uma página interna forçando a URL antes do login do usuário
Fonte de estímulo	Usuário

Ambiente	Produção, carga normal
Artefato	Qualquer página interna
Resposta	O sistema deve redirecionar o usuário para a tela de login
Medida da resposta	O sistema não pode permitir o acesso a páginas não autorizadas

3.3. Restrições Arquiteturais

Para evitar erosão arquitetural e o desalinhamento entre código e a arquitetura, listamos os domínios específicos que permitem definir regras arquiteturais que devem ser obedecidas na implementação do sistema.

- O sistema deve ser desenvolvido utilizando a linguagem de programação JAVA;
- O sistema deve ter arquitetura modular e orientada a serviços;
- O sistema deve ser modular e implantável em módulos, para facilitar a implantação de acordo com a prioridade e necessidade da empresa;
- O sistema deve ser hospedado em nuvem e/ou no data center da empresa(on-premise). Podendo ser total ou parcial em ambiente de nuvem e opcionalmente parcial em ambiente local;
- O sistema deve possuir seus serviços acessados apenas pelas instâncias que rodem o api gateway;
- A implantação do sistema deve ser realizada de forma automatizada utilizando mecanismos de integração contínua e automação de testes. O sistema deve ser implantado utilizando recursos de integração contínua;
- O sistema deve ter interfaces para integração com sistemas externos. O sistema deve realizar integração com APIs de terceiros;
- O sistema deve possuir seu layout responsivo, devendo abrir de forma adaptativa em qualquer resolução(celular, desktop e tablet). Podendo ser utilizado com facilidade tanto em computadores como dispositivos móveis;
- O sistema deve conter automação de testes no pipeline de integração contínua.
- As integrações entre os sistemas devem utilizar autenticação de segurança.

3.4. Mecanismos Arquiteturais

Mecanismo de Análise	Mecanismo de Design	Mecanismo de Implementação
Comunicação entre processos	Contêiner Web e Aplicação	Docker
Integração com outros módulos e/ou sistemas	Interfaces utilizando XML e/ou JSON	WebServices e WebAPI
Mensageria	Integração através de mensagens	RabbitMQ
Documentação e teste de API	Ciclo de desenvolvimento de uma API REST	Spring Swagger
Segurança	Framework de autenticação e autorização	Spring Security
Log	Framework de Log	log4J
Persistência	Framework ORM	Hibernate ORM
Persistência	Banco de dados relacional	MYSQL
Build	Geração de artefato para servidor de aplicação	Maven
Deploy	Deploy da aplicação no servidor e testes automatizados.	AWS CodeDeploy
CI/CD	Pipeline de integração contínua	AWS CodeDeploy

Front-End	Interface de comunicação com o usuário do sistema.	HTML5, bootstrap
Versionamento	Versionamento do código-fonte da aplicação.	Git
Ambiente de implantação (Host)	Servidor de aplicação	AWS EC2
Testes	Testes de unidade	JUnit
Autenticação e Autorização	Verificação das credenciais e tentativas de conexão.	OAuth
Notificação	Envio de e-mails de alerta disparados pelo sistema	Java Mail
Alta disponibilidade	Balanceamento de carga dos serviços.	Spring Cloud Netflix: Eureka, Zuul
Descoberta	Registro de serviços.	Eureka
Sistema Operacional	Sistema que será executado nos servidores	Linux

4. Modelagem e projeto arquitetural

4.1. Modelo de casos de uso

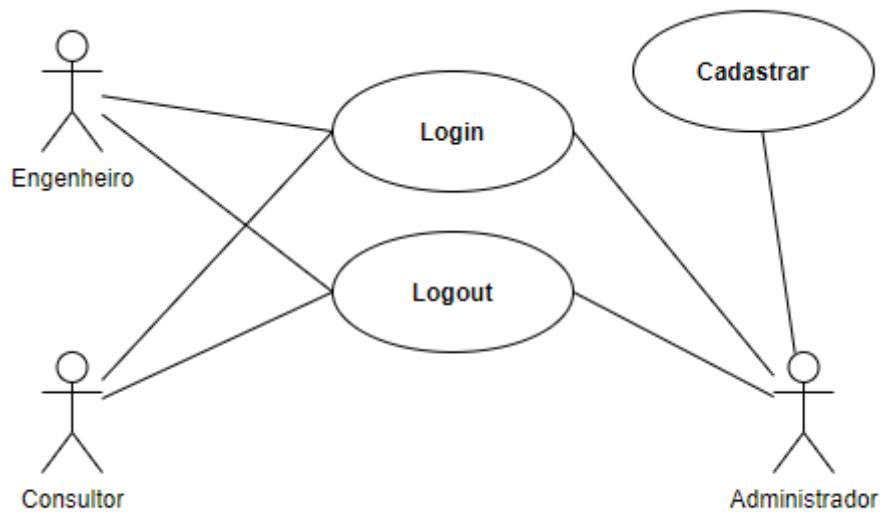


Figura 2 - Módulo de controle de usuários

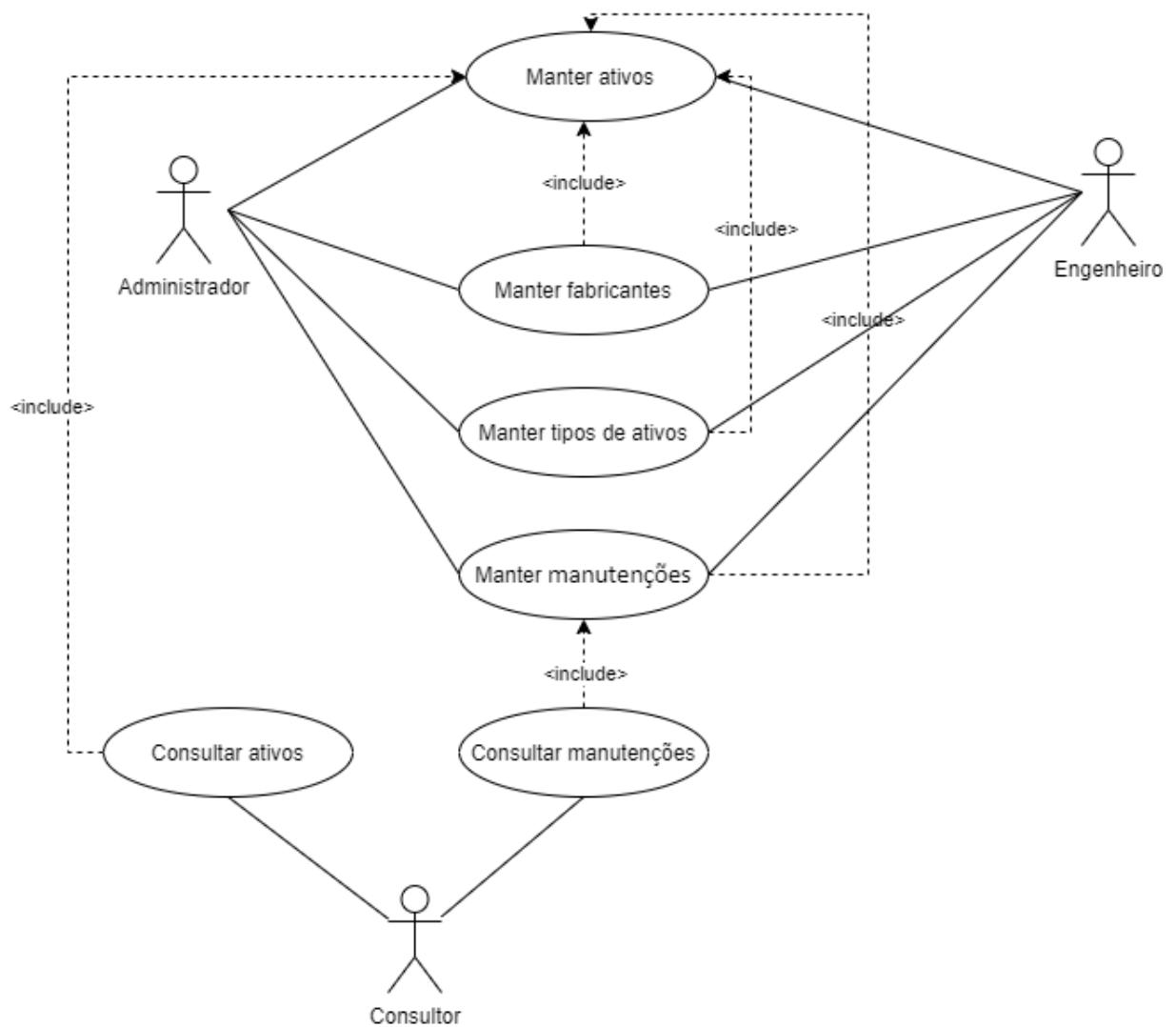


Figura 3 - Módulo de controle de ativos

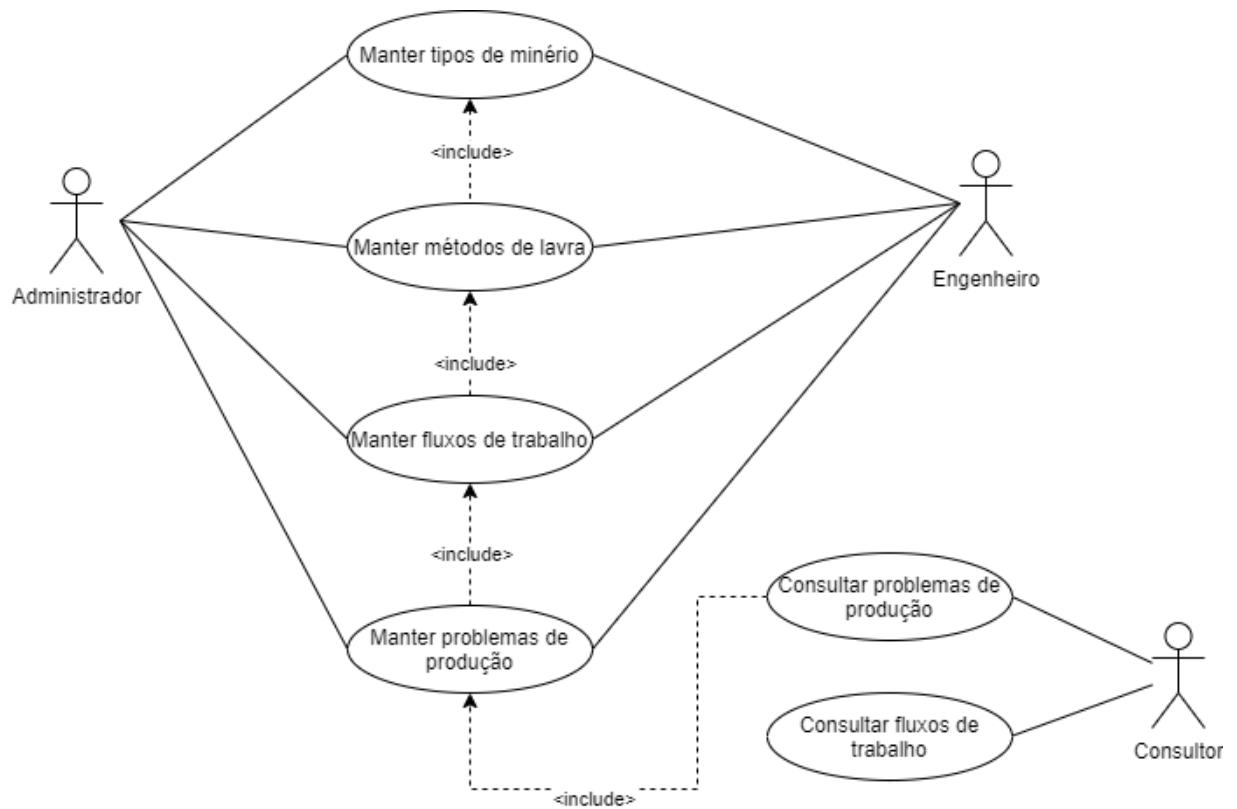


Figura 4 - Módulo de processos minerários

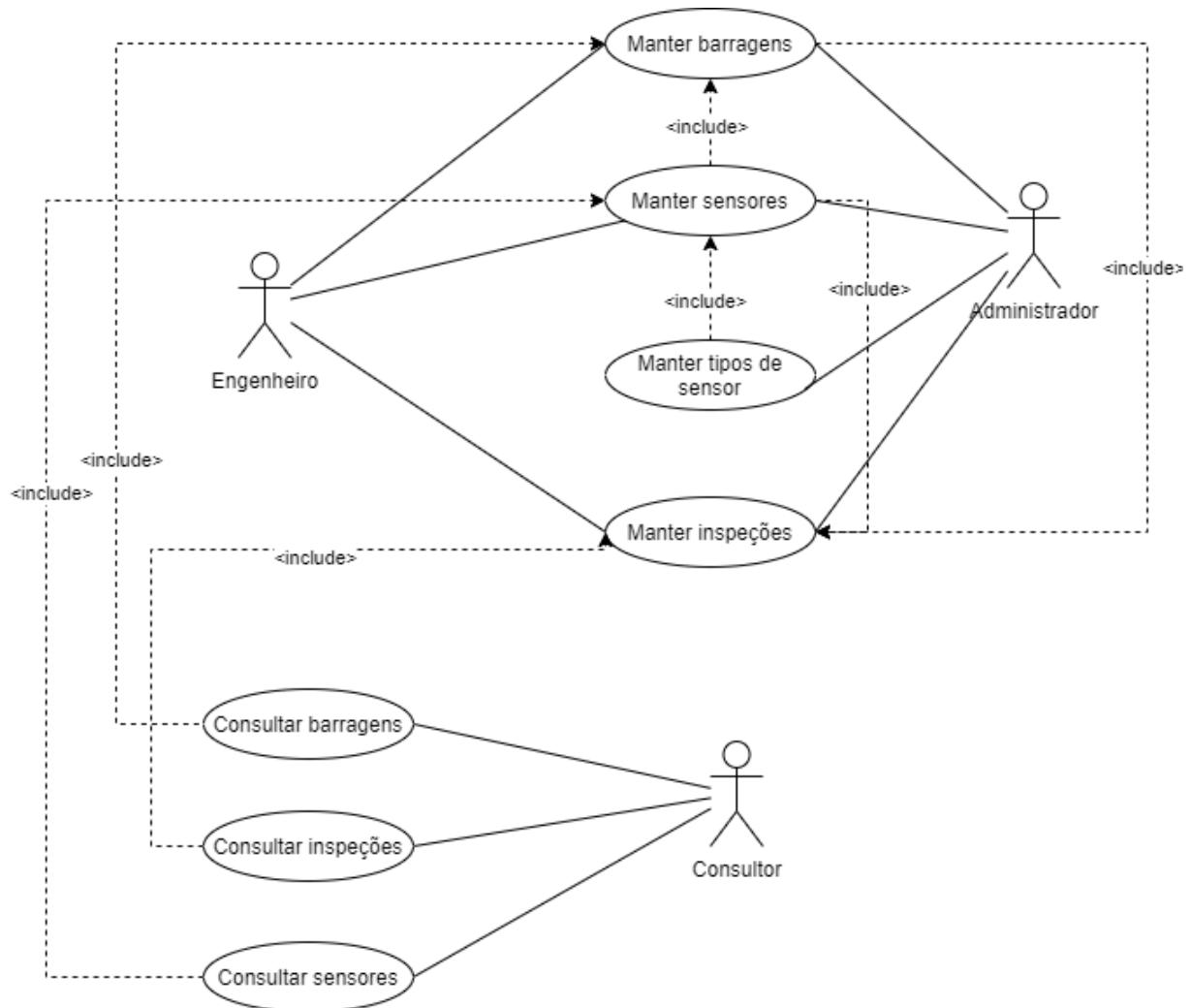


Figura 5 - Módulo de monitoramento de barragens

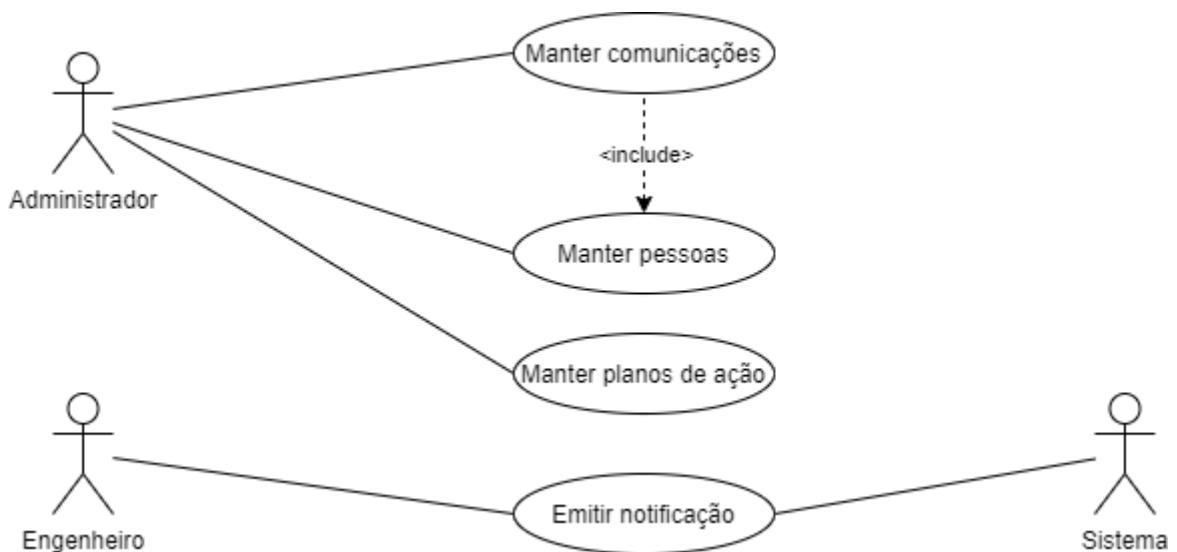


Figura 6 - Módulo de segurança e comunicação

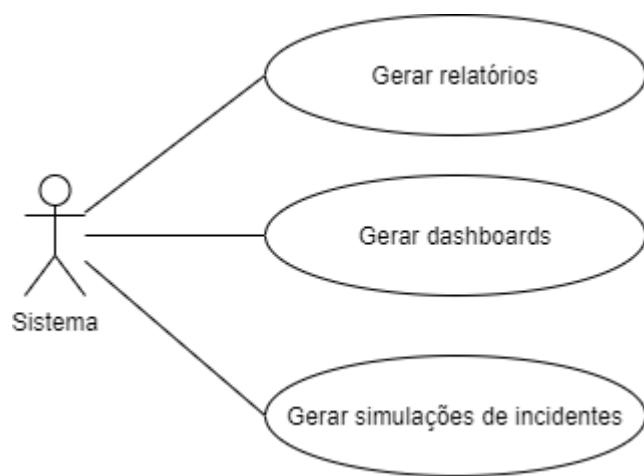


Figura 7 - Módulo de inteligência do negócio (BI)

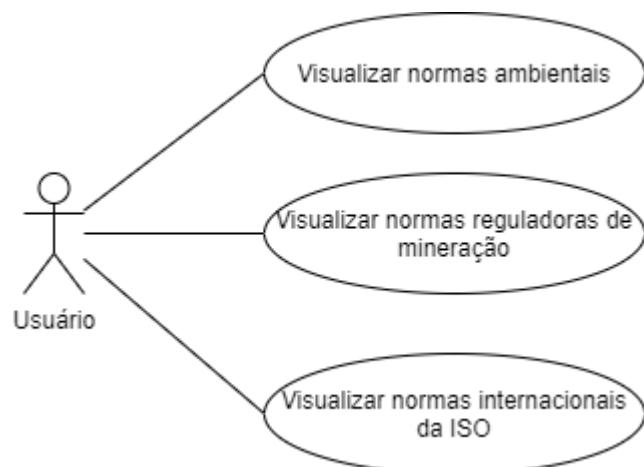


Figura 8 - Módulo de compliance

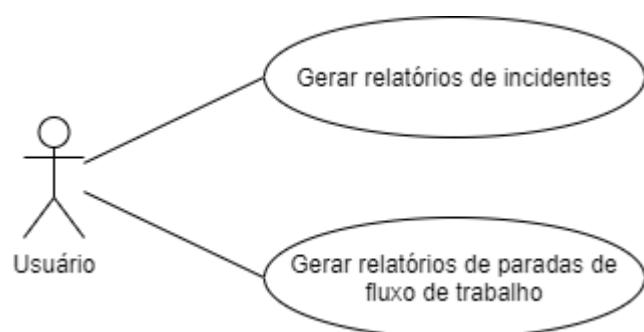


Figura 9 - Módulo de relatórios de acompanhamento

4.2. Modelo de componentes

O diagrama de componentes ilustra como as classes deverão ser organizadas dentro das tecnologias e arquitetura do sistema. Essa perspectiva primariamente proporciona suporte ao gerenciamento da configuração das partes do sistema, que foi formada pelos componentes arquiteturais, que por sua vez, foram reunidos de várias maneiras para produzir interfaces bem definidas de acordo com suas responsabilidades.

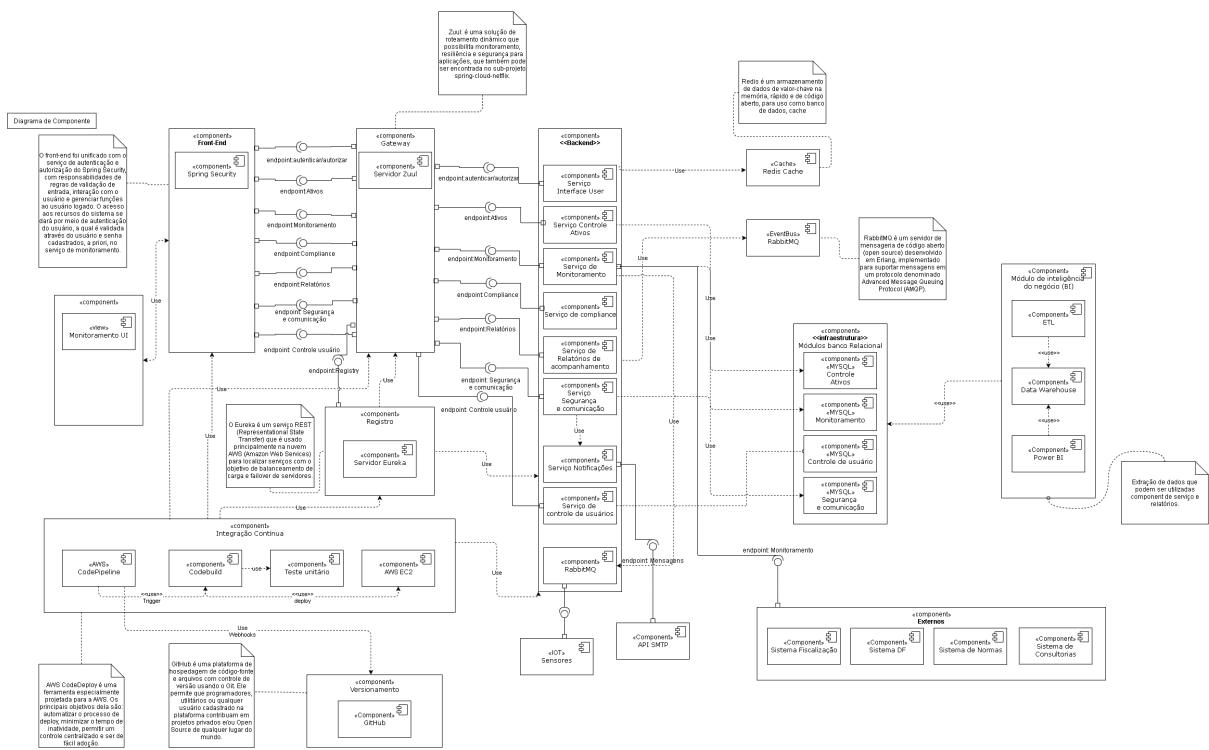


Figura 10 - Diagrama de Componentes

A arquitetura proposta, funciona com a separação da aplicação em duas frentes principais, uma com front-end com toda estruturação de um aplicativo Web composto por HTML, Bootstrap e CSS, Spring MVC, e um back-end em módulos portados por um conjunto de serviços independentes. Os serviços funcionam de forma independente dos demais e possuem sua própria base de dados. Os dados fornecidos por esses serviços, são afunilados em apenas um servidor Gateway, que quando chamados, irão fornecer os dados e exibir as informações em um navegador de mercado.

O front-end foi unificado com o serviço de autenticação e autorização do Spring Security, com responsabilidades de regras de validação de entrada, interação com o usuário e gerenciar funções ao usuário logado. O acesso aos recursos do sistema se dará por meio de autenticação do usuário, a qual é validada através do usuário e senha cadastrados, a priori, no serviço de monitoramento.

Os serviços são registrados em um servidor de registro, garantindo sempre o acesso às instâncias de forma dinâmica e com balanceamento de carga. Para gerenciar os endpoints descobertos anteriormente, e linkar aos pontos mais adequados, a comunicação será realizada unicamente através uma API Gateway, filtrando o tráfego no momento da integração, e sendo acessível somente por autenticação. O funcionamento do proxy de APIs são divididos em três papéis distintos:

- Spring Cloud: Responsável por integrar todas as soluções em aplicações Spring Boot, onde toda configuração é realizada por anotações e propriedades do application.properties;
- Eureka: Responsável por registrar as aplicações;
- Zuul: Responsável por ser a porta de entrada das chamadas e direcionar as chamadas para as aplicações registradas no Eureka.

O sistema também prevê a integração com serviços externos. Interfaces externas são requeridas para serem acionadas por interfaces internas, acionando envio de e-mails e consultas de normas ambientais, para isso o sistema respeita a interface exposta de cada um desses serviços.

O código fonte no GitHub, utilizando-se do padrão Git, será integrado ao AWS CodeDeploy, onde se prevê o deploy automático com os artefatos transformados em imagens utilizadas em Docker containers. Será construído o pipeline de integração contínua para construção dos artefatos usando o AWS Pipeline, e execução de testes unitários com o JUnit utilizando AWS CodeBuild. O versionamento dos artefatos criados, serão alocados no AWS S3.

O sistema também prevê a integração com o protocolo MQTT, que viabiliza o enfileiramento avançado de mensagens que são transmitidas dos sensores instalados ao longo da barragem, para o receptor publicado dentro do módulo de monitoramento. para transmissão dos dados de monitoramento utilizados nos sensores instalados ao longo das barragens. O sensor publica a mensagem dentro da mensageria, e o sistema ouve e grava logo assim que possível.

O sistema também prevê a integração com a ferramenta Redis para uso do cache. Utilizado para cachear em memória e diminuir o tráfego de informações solicitadas utilizadas com frequência.

4.3. Modelo de implantação

Toda infraestrutura será distribuída utilizando-se de containers Docker, com seu uso provendo a cada build um novo servidor já instanciado com a aplicação instalada. Com o uso de micro serviços, cada módulo disponível será instanciado em uma máquina isolada, não interferindo no funcionamento dos demais módulos. Dessa forma, possibilita uma fácil implantação do sistema, sendo essa dividida seguindo a divisão de módulos do sistema, de forma a permitir a implantação de acordo com a prioridade e necessidade do ecossistema da organização.

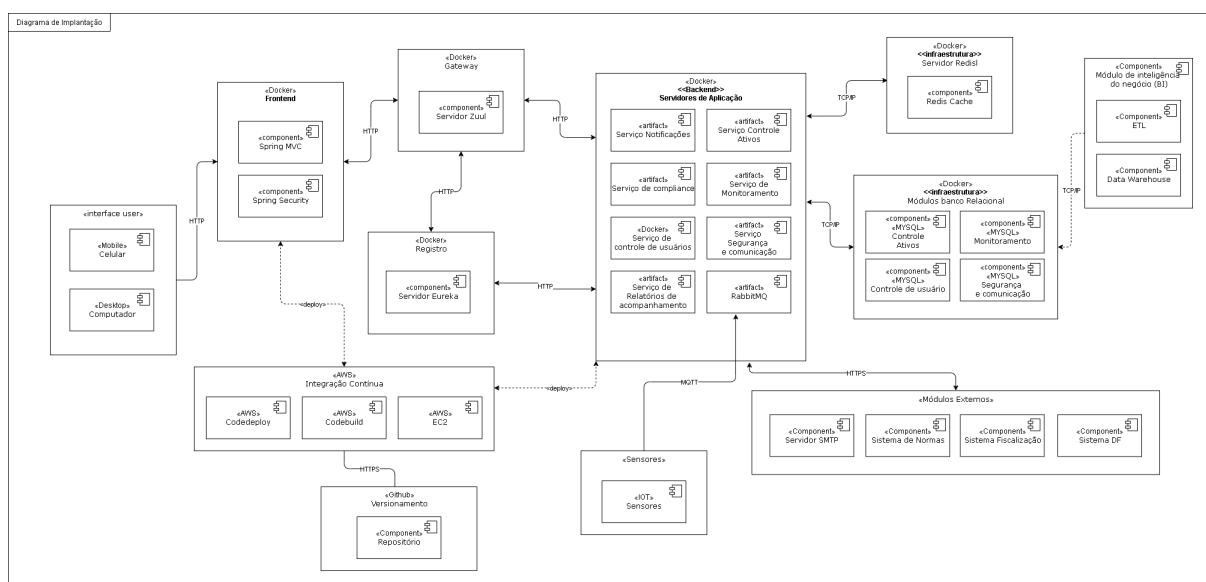


Figura 11 – Diagrama de implantação

Abaixo representamos o descriptivo da implantação dos servidores e seus artefatos, com detalhamentos da estrutura e seus sistemas explicadas no modelo de Diagrama de Implantação.

Componente	Descrição
Cliente Browser	Representa em qual tipo de interface o usuário utilizará para acessar e interagir com o sistema. Pode ser utilizado a partir de um

	computador pessoal ou dispositivo móvel.
Servidor WEB	Responsável por disponibilizar a autenticação/autorização, além de páginas da aplicação que serão acessadas pelos usuários através de um navegador.
Servidor de aplicação	Responsável por prover toda a infraestrutura necessária para que a aplicação funcione no servidor.
Servidor de Versionamento.	Responsável por alocar todos os arquivos de controle de versão distribuído.
Servidor de integração contínua	Responsável pelo build, test e deploy da aplicação nos servidores de produção.
Servidor de Gateway	Responsável por todas as requisições e roteamento dinâmico para aplicações nos servidores de produção.
Servidor de cache	Responsável por manter os dados e processos mais acessadas, armazenados em memória.
Servidor de banco de dados relacional	Representa o servidor responsável por armazenar e gerenciar os dados nos serviços da aplicação.
Servidor de Módulo de inteligência do negócio (BI)	Componente responsável por transportar os dados de todos os bancos de dados da aplicação para um servidor centralizado de inteligência do negócio.
Servidores Externos(Backend)	Representa os servidores acessados externamente por requisições HTTP.

4.4. Modelo de dados

Cada módulo de serviço tem seu banco de dados exclusivo ao invés de utilizar um único banco de dados para toda aplicação. O não compartilhamento de tabelas entre serviços, aumenta a coesão e diminui o acoplamento, uma vez que alterações feitas em tabelas não afetarão o modelo de dados de outros serviços e alterações não precisarão ser replicadas em vários locais.

Para manter o sistema com uma performance aceitável. A POC implementando nesse documento, não possui desacoplamento dos bancos de dados. Cada módulo tem um modelo de dados de domínio compartilhado no mesmo contexto, porém com representações internas que façam mais sentido para cada um dos serviços.

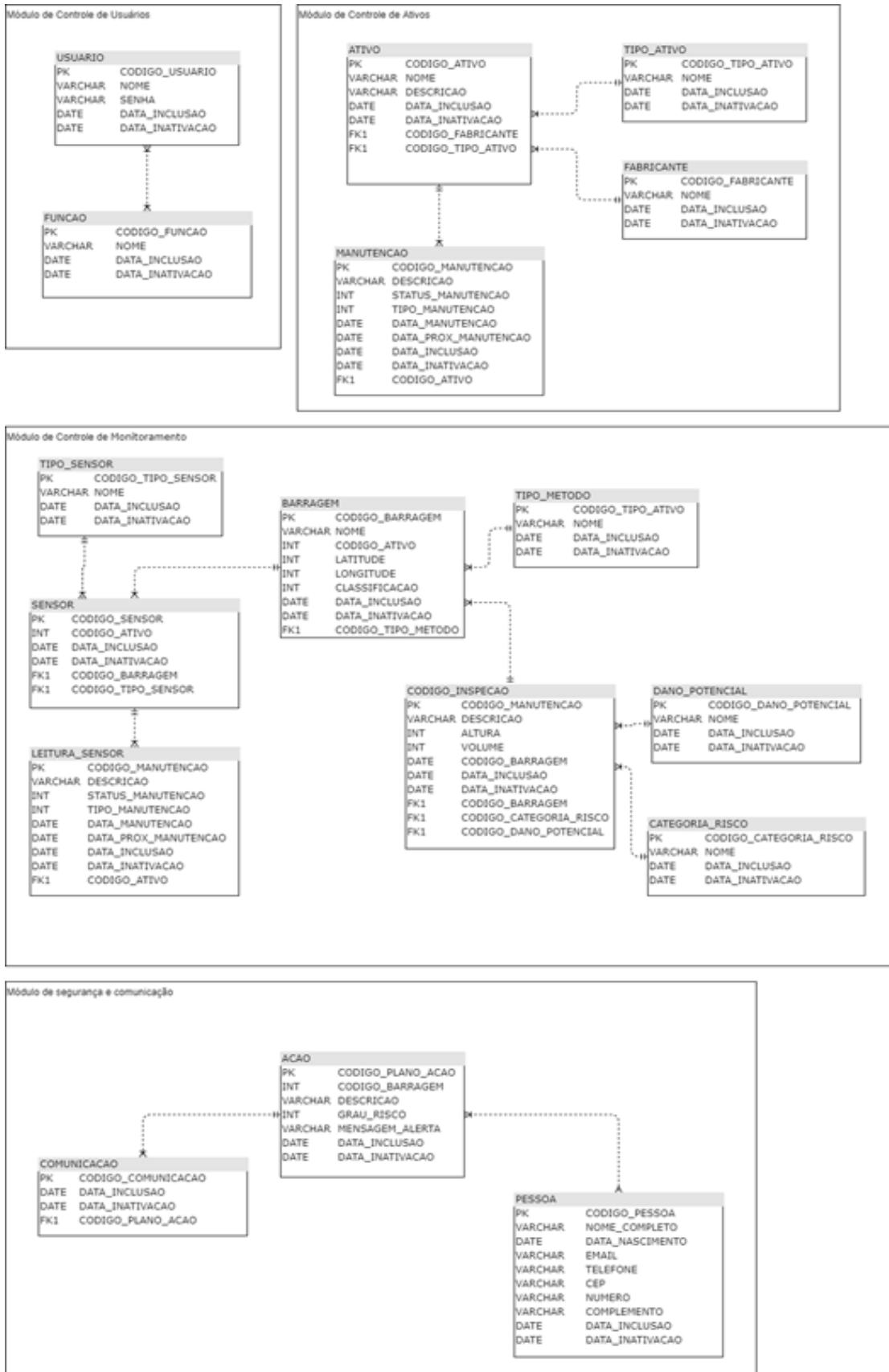


Figura 12 – Diagrama de Modelo de Dados

5. Prova de Conceito (POC) / protótipo arquitetural

5.1. Implementação e Implantação

A prova de conceito tem como objetivo validar se alguns dos requisitos não funcionais foram atendidos. Ela contempla os seguintes módulos: módulo de controle de usuários, módulo de controle de ativos, módulo de monitoramento de barragens e módulo de segurança e comunicação.

Nesta prova de conceito, foram validados os seguintes requisitos não funcionais:

- **Segurança:** O sistema deve ter alta segurança de seus módulos, disponibilizando o acesso somente aos usuários autorizados.

Este requisito não funcional (RNF) foi escolhido devido à segurança do sistema, evitando o acesso a dados sensíveis por pessoas não autorizadas.

Os critérios de aceite são:

- Impedir o acesso sem autenticação às páginas internas
- Redirecionar o usuário para a tela de login caso tente acessar uma página interna sem estar autenticado.

- **Usabilidade:** O sistema deve possuir o layout responsivo, com a finalidade de ter o acesso possível via desktop ou dispositivos móveis.

Este RNF foi escolhido pela necessidade de o sistema ser compatível com vários tipos de dispositivos, resoluções e tamanhos de tela.

Os critérios de aceite são:

- O sistema deve suportar várias resoluções de tela, se adequando e adaptando os componentes quando necessário.
- O sistema deve manter o padrão visual em diferentes resoluções.

- O sistema deve ser compatível com os principais browser do mercado como: Microsoft Edge, Chrome e Firefox.
- **Desempenho:** O sistema deve possuir respostas rápidas as ações do usuário.

Este RNF foi escolhido para garantir uma boa performance e desta forma não comprometer a utilização do sistema

Os critérios de aceite são:

- O tempo de renderização de tela não deve exceder 5 segundos.
- O retorno de uma ação do usuário não deve exceder 5 segundos.
- **Interoperabilidade:** O sistema deve possuir integração com serviço externo para o envio de emails em caso de notificações de incidentes.

Este RNF foi escolhido para garantir o funcionamento da notificação em caso de incidentes, um item essencial do sistema.

O critério de aceite é:

- Ao ocorrer um incidente devido a leitura de sensores ou um pedido de evacuação do engenheiro, um email deve ser enviado às pessoas responsáveis de acordo com o plano de ação.

5.1.1. Tecnologias utilizadas

Caso de uso	Tecnologias
Interface do Usuário(Frontend)	Bootstrap, Jquery, CSS, HTML, Spring MVC, Spring Security, MySQL
Manter Ativos	Spring Boot, Spring data, MySQL,

Manter Barragens	Spring cloud, Zuul e Eureka
Manter Classificações de Risco	
Manter Sensores	
Manter Comunicações	
Manter Planos de Ação	
Manter Notificações	
Monitorar Barragens	Spring Boot, Spring data, MySQL, RabbitMQ, Spring cloud, Zuul e Eureka
Receber Leituras dos Sensores	RabbitMQ
Acionar Planos de Ação	Spring Boot, MySQL, RabbitMQ

5.2. Interfaces/ APIs

Sessão 1: Interface de Mensageria para fila de leitura dos sensores

O Sistema de Mensageria utiliza de interfaces HTTP para efetuar uma troca de informações desacoplada do sistema de monitoramento. Os medidores utilizam de acesso direto ao barramento para inserir os dados em sua fila de processos.

Toda comunicação entre o sistema e os sensores, é realizada pela SDK implementada no Java através das suas dependências instaladas no pom.xml..O protocolo de comunicação em rede utilizado pela mensageria será o AMQP, que coordena o envio e a recepção de mensagens em uma fila.

API	RabbitMQ HTTP API	
URL	http://localhost:15672/api/exchanges/%2f/amq.default/publish	
Verbo HTTP	POST	
Headers	Content-Type	application/json

	Accept	/*
	Accept-Encoding	gzip, deflate, br
	Authorization	Basic Auth
	Username	guest
	Password	guest
Requisição	<pre>curl --location --request POST 'http://localhost:15672/api/exchanges/%2f/amq.default/publish' \ --header 'Content-Type: application/json' \ --header 'Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=' \ --data-binary '{ "properties": { "content_type": "text/plain" }, "routing_key": "OrderLeituraSensorQueue", "payload": "7/23.42", "payload_encoding": "string" }'</pre>	
Resposta	<pre>{"routed":true}</pre>	

The screenshot shows a Postman interface with a POST request to `http://localhost:15672/api/exchanges/%2f/amq.default/publish`. The **Body** tab is selected, displaying the following JSON payload:

```

1 "properties": {
2   "content_type": "text/plain"
3 },
4 "routing_key": "OrderLeituraSensorQueue",
5 "payload": "7/23.42",
6 "payload_encoding": "string"
7
8
  
```

The response section shows a green **200 OK** status with a **30 ms** duration and **296 B** size. Below the response, there are buttons for **Pretty**, **Raw**, **Preview**, and **Visualize**.

Figura 13 – exemplo de request e response da Mensageria para fila de leitura dos sensores

Campos retornados pelo serviço de mensageria:

Campo	Tipo	Descrição
<code>routed</code>	boolean	Identifica com sucesso o envio da mensagem ao barramento. (true or false)

Sessão 2: Interface inserção de leituras ao sensor

O sistema expõe uma interface para receber requisições dos sensores cadastrados previamente. A API tem como retorno padrão uma mensagem com dados do evento carimbado com hora do recebimento

API	Monitoramento Leitura Sensor HTTP API
URL	<code>http://localhost:8765/api/monitoramento/sensor/leitura</code>
Verbo HTTP	POST

Headers	Content-Type	application/json
	Accept	*/*
	Accept-Encoding	gzip, deflate, br
	Authorization	Basic Auth
	Username	zuul
	Password	zuul
Requisição	<pre>curl -X POST "http://localhost:8061/sensor/leitura" -H "accept: */*" -H "Content-Type: application/json" -d " { "leitura": 123, "sensor": { "codigo": 1, "codigoAtivo": 1 } }</pre>	
Resposta	<pre>{ "codigo": 0, "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "leitura": 0, "sensor": { "barragem": { "classificacao": "A", "codigo": 0, "codigoAtivo": 0, "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "latitude": 0, "longitude": 0, "nome": "string", "tipoMetodo": { "codigo": 0, "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "nome": "string" } } } }</pre>	

```

        }

    },
    "codigo": 0,
    "codigoAtivo": 0,
    "dataInativacao": "dd/MM/yyyy HH:mm:ss",
    "dataInclusao": "dd/MM/yyyy HH:mm:ss",
    "tipoSensor": {
        "codigo": 0,
        "dataInativacao": "dd/MM/yyyy HH:mm:ss",
        "dataInclusao": "dd/MM/yyyy HH:mm:ss",
        "maxLeitura": 0,
        "minLeitura": 0,
        "nome": "string"
    }
}
}

```

POST <http://localhost:8765/api/monitoramento/sensor/leitura>

Params Authorization Headers (13) **Body** Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** [Beautify](#)

```

1 {
2     "leitura": 1233,
3     "sensor": {
4         "codigo": 1,
5         "codigoAtivo": 1
6     }
7 }

```

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 88 ms Size: 591 B Save Response

Pretty Raw Preview Visualize JSON [Beautify](#)

```

1 {
2     "codigo": 480,
3     "sensor": {
4         "codigo": 1,
5         "codigoAtivo": 1,
6         "tipoSensor": null,
7         "dataInclusao": null,
8         "dataInativacao": null,
9         "barragem": null
10    },
11    "leitura": 1233.0,
12    "dataInclusao": "25/11/2020 03:00:24",
13    "dataInativacao": null
14 }

```

Figura 14 – exemplo de request e response da inserção de leitura.

Campos utilizados para executar uma nova inserção de leitura:

Campo	Obrigatório	Tipo	Descrição
leitura	Sim	Double	Leitura enviada pelo Sensor
sensor.codigo	Sim	Long	Código do Sensor
sensor.codigoAtivo	Sim	Long	Código do Ativo

Campos retornados da consulta para nova inserção de leitura:

Campo	Tipo	Descrição
codigo	Double	Número sequencial da leitura
sensor	Object	Sensor da leitura
leitura	Long	Código do Ativo
dataInclusao	Data	Data de Inclusão
dataInativacao	Data	Data de Inativação

Sessão 3: Interface de Comunicações

O sistema expõe um método para retornar as comunicações dos sensores e barragens relacionadas a não conformidades de evacuações e métricas encontradas durante os diversos processos do sistema, permitindo o retorno de todos os incidentes ocorridos ao longo.

API	Segurança e Comunicação API	
URL	http://localhost:8765/api/seguranca/comunicacao	
Verbo HTTP	GET	
Headers	Content-Type	application/json
	Accept	*/*

	Accept-Encoding	gzip, deflate, br
	Authorization	Basic Auth
	Username	zuul
	Password	zuul
Requisição	<pre>curl -X GET "http://localhost:8060/comunicacao" -H "accept: application/json"</pre>	
Resposta	<pre>[{ "codigo": 0, "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "planoAcao": { "codigo": 0, "codigoAtivo": 0, "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "descricao": "string", "grauRisco": "Baixo", "mensagemAlerta": "string", "pessoas": [{ "cep": "string", "codigo": 0, "complemento": "string", "dataInativacao": "dd/MM/yyyy HH:mm:ss", "dataInclusao": "dd/MM/yyyy HH:mm:ss", "dataNascimento": "dd/MM/yyyy HH:mm:ss", "email": "string", "nomeCompleto": "string", "numero": "string", "telefone": "string" }] }]]</pre>	

The screenshot shows a Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:8765/api/seguranca/comunicacao
- Headers:** (9) - Includes Authorization, Content-Type (application/x-www-form-urlencoded), and Content-Length.
- Body:** (x-www-form-urlencoded)

Key	Value	Description
Key	Value	Description
- Test Results:** Status: 200 OK, Time: 77 ms, Size: 113.61 KB
- Response Body (Pretty JSON):**

```

1
2   {
3     "codigo": 1,
4     "planoAcao": [
5       {
6         "codigo": 1,
7         "pessoas": [
8           {
9             "codigo": 1,
10            "nomeCompleto": "Vicente Benjamin Bryan da Rosa",
11            "dataNascimento": "01/11/1988 02:00:00",
12            "email": "carloscalebmonteiro_gvmail.br",
13            "telefone": "(61) 2744-6935",
14            "cep": "73753-074",
15            "numero": "30",
16            "complemento": "Quadra 5 MC",
17            "dataInclusao": "01/11/2020 03:00:00",
18            "dataInativacao": null
19          },
20          {
21            "codigo": 2,
22            "nomeCompleto": "Raul Bruno da Luz",
23            "dataNascimento": "01/11/1988 02:00:00",
24            "email": "rrebecaauroranunes@vanguarda.tv",
25            "telefone": "(61) 2744-6935",
26            "cep": "73753-074",
27            "numero": "30",
28            "complemento": "Quadra 5 MC".
29          }
30        ]
31      }
32    }
33  
```

Figura 15 – exemplo de request e response da comunicação.

Campos retornados na execução da interface de comunicação:

Campo	Tipo	Descrição
codigo	Long	Código sequencial da comunicação.
planoAcao	Array(Object)	Plano de ação cadastrado para determinada comunicação.
pessoas	Array(Object)	Lista de pessoas cadastradas para determinado plano de ação.

dataInclusao	Data	Data de Inclusão.
dataInativacao	Data	Data de Inativação.

6. Avaliação da Arquitetura

6.1. Análise das abordagens arquiteturais

A análise propõe a utilização de uma arquitetura de referência baseado no uso de micro serviços, utilizando a plataforma open source Java Enterprise Edition (Java EE) e Spring Boot como framework, que corresponde a uma infraestrutura de componentes e serviços para o desenvolvimento e execução de módulos independentes e específicos a sua função. Uma vez que as aplicações não são acopladas, poderemos implantar o sistema utilizando uma diversidade enorme de métodos, incluindo com base em imagens e utilizando o Docker para tecnologia de conteinerização e Docker Compose como orquestrador de containers. As atividades e as etapas deste método de arquitetura abordado foram exemplificadas de forma integrada como parte do processo de desenvolvimento e centradas na arquitetura de software. Um protótipo de desenvolvimento (POC), é utilizado para exemplificar a abordagem proposta.

6.2. Cenários

Cenário 1: O sistema deve garantir que o usuário tenha acesso apenas as funcionalidades que ele esteja autorizado. Ao tentar acessar uma página privada por meio de URL direta antes da autenticação ou uma página que seu perfil não possui autorização, o sistema deve redireciona-lo para a tela de login e informa-lo que não acesso não foi autorizado. Desta forma é garantida a segurança do sistema e atendido o requisito não funcional.

Cenário 2: As telas do sistema devem se adaptar a diferentes tipos de dispositivos e resoluções. Ao utilizar o browser de um tablet, por exemplo, para acessar a aplicação os componentes de tela devem ser repositionados para não comprometer a usabilidade e, assim, atender ao requisito não funcional.

Cenário 3: O sistema deve possuir um bom desempenho não excedendo o tempo de resposta de 5 segundos após uma ação do usuário. A garantia mínima deste desempenho atende o requisito não funcional.

Cenário 4: Ao ocorrer uma notificação de incidente, um e-mail deve ser disparado pelo módulo de segurança e comunicação de acordo com o que foi pré-definido no plano de ação. A envio correto deste e-mail é de suma importância para o alerta de incidentes e garante que o requisito não funcional foi atendido.

Na priorização foi utilizado o método de Árvore de Utilidade reduzida e com prioridades. Foi categorizado de acordo os atributos de qualidade a que estão relacionados e então classificados em função de sua importância e complexidade, considerando a percepção de negócio e arquitetura. As duas variáveis de priorização "Importância" e "Complexidade", apresentadas nas colunas IMP. e COM. respectivamente forma classificadas em alta (A), média (M) e baixa (B) de acordo com as características do requisito.

Atributos de Qualidade	Cenários	IMP.	COM.
Segurança	Cenário 1: O sistema deve possuir alta segurança, disponibilizando o acesso somente aos usuários autorizados	A	A
Acessibilidade	Cenário 2: O sistema deve suportar diferentes tipos de dispositivos	M	B
Desempenho	Cenário 3: O sistema deve possuir bom desempenho	M	A
Interoperabilidade	Cenário 4: Envio de email em caso de alerta de incidente	A	M

6.3. Avaliação

Cenário 1:

Atributo de Qualidade:	Segurança
Requisito de Qualidade:	O sistema deve possuir alta segurança, disponibilizando o acesso somente aos usuários autorizados
Preocupação:	
Impedir o acesso não autorizado a páginas internas do sistema	
Cenário(s):	
Cenário 1	
Ambiente:	

Produção, carga normal	
Estímulo:	
Usuário tentar acessar uma página privada (via URL direta) antes de se autenticar no sistema.	
Mecanismo:	
Interceptar as requisições validando se há um usuário autenticado e autorizado a visualizar a página.	
Medida de Resposta:	
Ao ter o pedido recusado, o usuário deve ser redirecionado à tela de login.	
Considerações sobre a arquitetura:	
Riscos	Haver falha na criação e validação de tokens de segurança.
Pontos de Sensibilidade:	Servidor operando em modo HTTPS
Tradeoff:	Não existe

Evidências do Cenário 1:

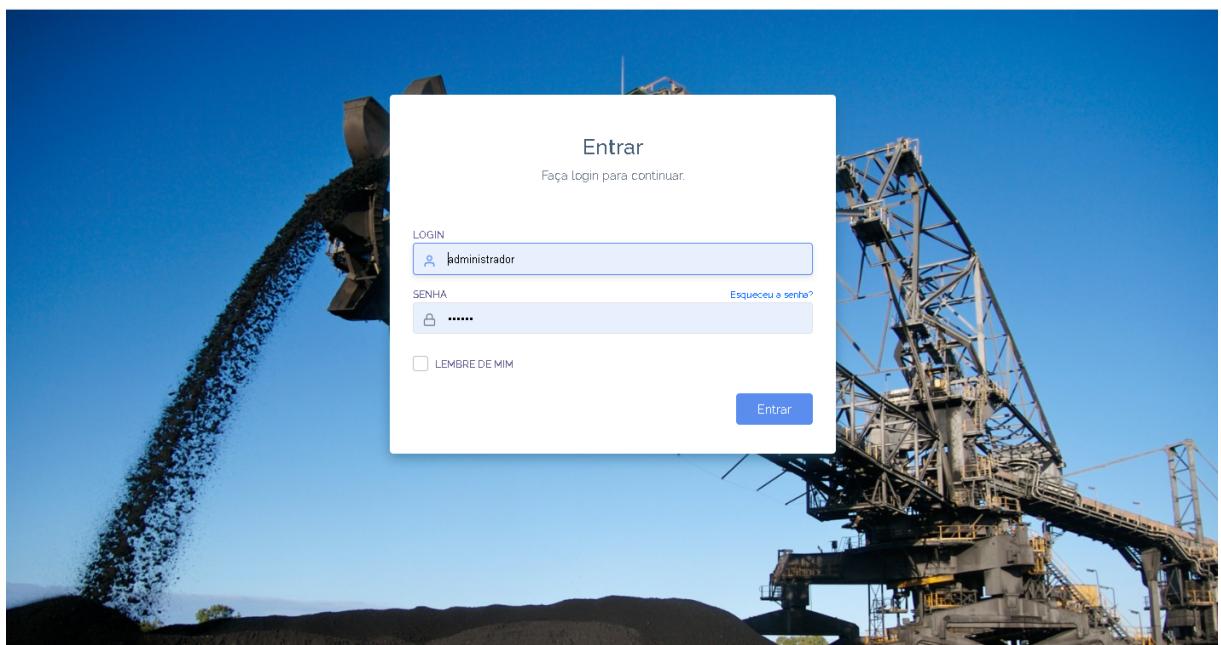


Figura 16 – Usuário não autenticado e sem autorização de acesso ao sistema.

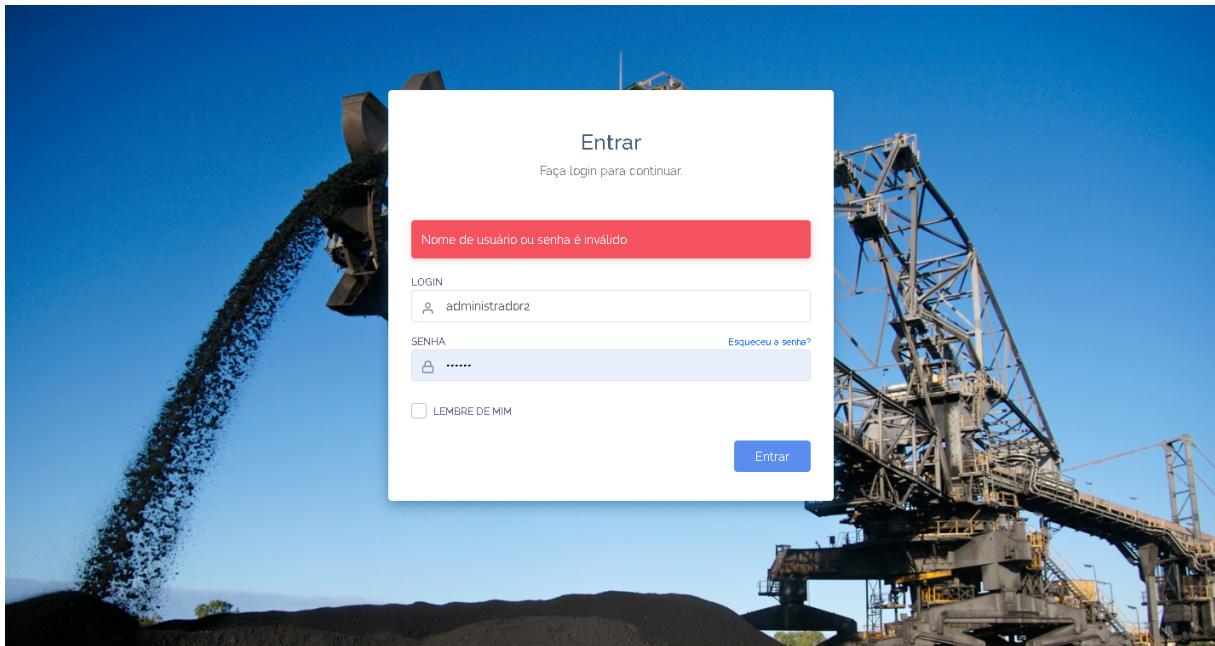


Figura 17 – Usuário não autenticado é redirecionado para tela de autenticação.

The dashboard includes the following sections:

- MENU:**
 - Painel Principal
 - Usuários
 - Cadastro de Ativos
 - Monitoramento
 - Segurança/Comunicação
 - Sair
- Home:** Brumadinho - Córrego do Feijão
- Localização:** Map showing the location of the monitoring site.
- Última Inspeção:**
 - apto
 - Categoria de Risco: Baixo
 - Dano Potencial: Baixo
 - Altura: 22,5
 - Volume total: 3000,0
 - Data da última Inspeção: 12/07/2020 00:00:00
- Última Evacuação:**
 - DESCRICAO 1
 - Grau: BAIKO
 - Mensagem de Alerta: MENSAGEM_ALERTA 1
 - Data de Evacuação: 01/11/2020 01:00:00
- Sensor 1 - Medição de Nível:** Two line graphs showing level measurements over time (230 AM to 230 AM) with thresholds at 82, 90, 10, and 76.

Figura 18 – Usuário autenticado é redirecionado para tela de home.

The screenshot shows a web browser window with the URL localhost:8080/barragem. The page title is "Lista de Barragem". On the left, there is a sidebar menu with sections like "Painel Principal", "Usuários", "Cadastro de Ativos", "Monitoramento" (which is expanded, showing "Barragens", "Sensores", "Tipo Sensor", and "Inspeção"), "Segurança/Comunicação", and "Sair". The main content area displays a table titled "Lista de Barragem" with columns: #, Descrição, Ativo, Localização, Método, Data Inclusão, and Data Inativação. There are four rows of data, each representing a dam. The first row has the description "Barragem de Rejeitos 1", location "Brumadinho - Córrego do Feijão", method "Etapa única", and inclusion date "01/11/2020 01:00:00". The second row has the description "Barragem de Rejeitos 2", location "Mariana - Fundão", method "Alteamento a jusante", and inclusion date "01/11/2020 01:00:00". The third row has the description "Barragem de Rejeitos 3", location "Mariana - Germano", method "Alteamento a montante", and inclusion date "01/11/2020 01:00:00". The fourth row is partially visible. A search bar is at the top right of the table area.

Figura 19 – Usuário não autenticado tentando acessar uma página que exige autorização.

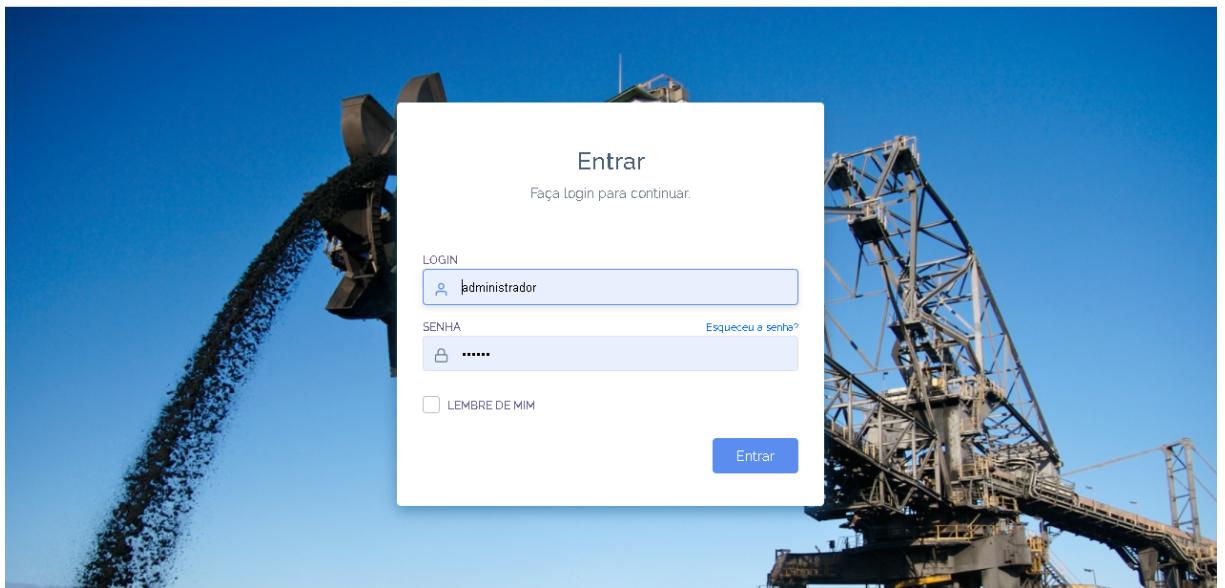


Figura 20 – Usuário não autenticado é redirecionado para tela de home.

Cenário 2:

Atributo de Qualidade:	Acessibilidade
Requisito de Qualidade:	O sistema deve suportar diferentes tipos de

	dispositivos
Preocupação:	As telas dos sistema devem se adequar a diferentes resoluções
Cenário(s):	Cenário 2
Ambiente:	Produção, carga normal
Estímulo:	Usuário utilizando o sistema a partir de um dispositivo móvel.
Mecanismo:	Layout responsivo e com o mínimo de elementos necessários.
Medida de Resposta:	Não há comprometimento na usabilidade sem perda de funções do sistema
Considerações sobre a arquitetura:	
Riscos	Layout que não se adequam à diferentes resolução tem o risco de impedir que o usuário consiga utilizar ou visualizar as funcionalidades do sistema, comprometendo a usabilidade
Pontos de Sensibilidade:	Não existe
Tradeoff:	Não existe

Evidências do Cenário 2:

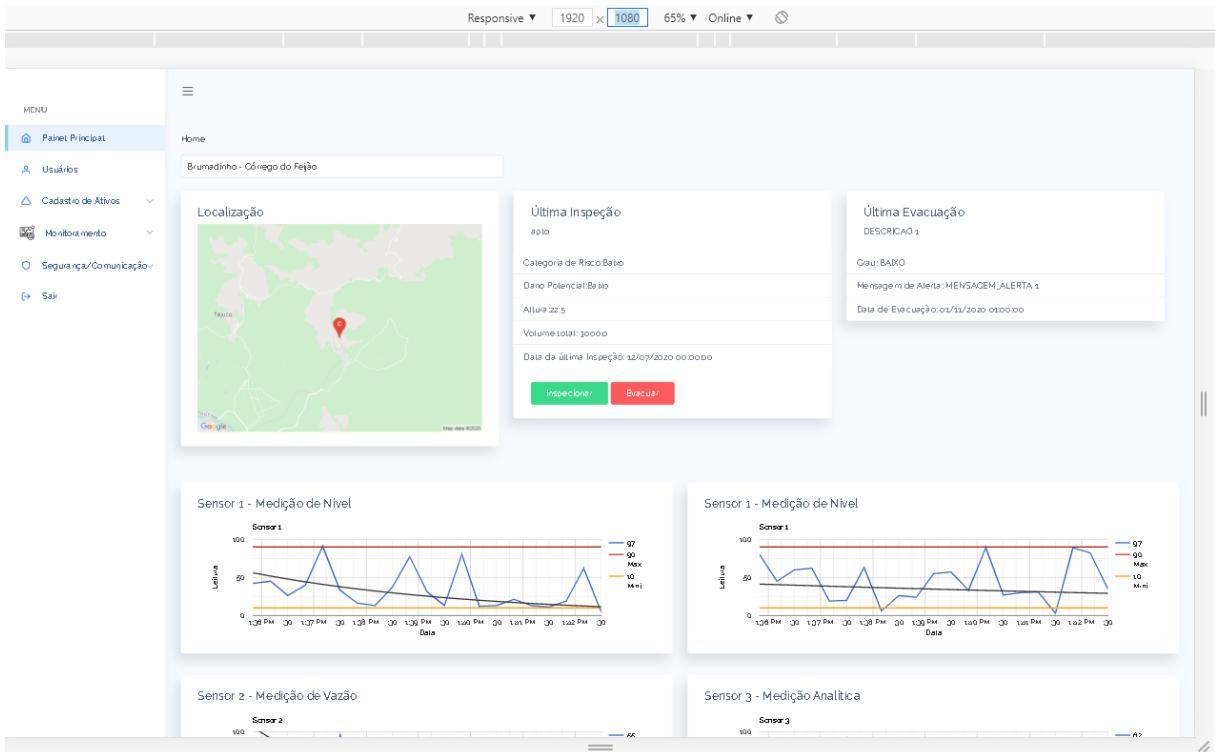


Figura 21 – Resolução 1920x1080 Desktop.

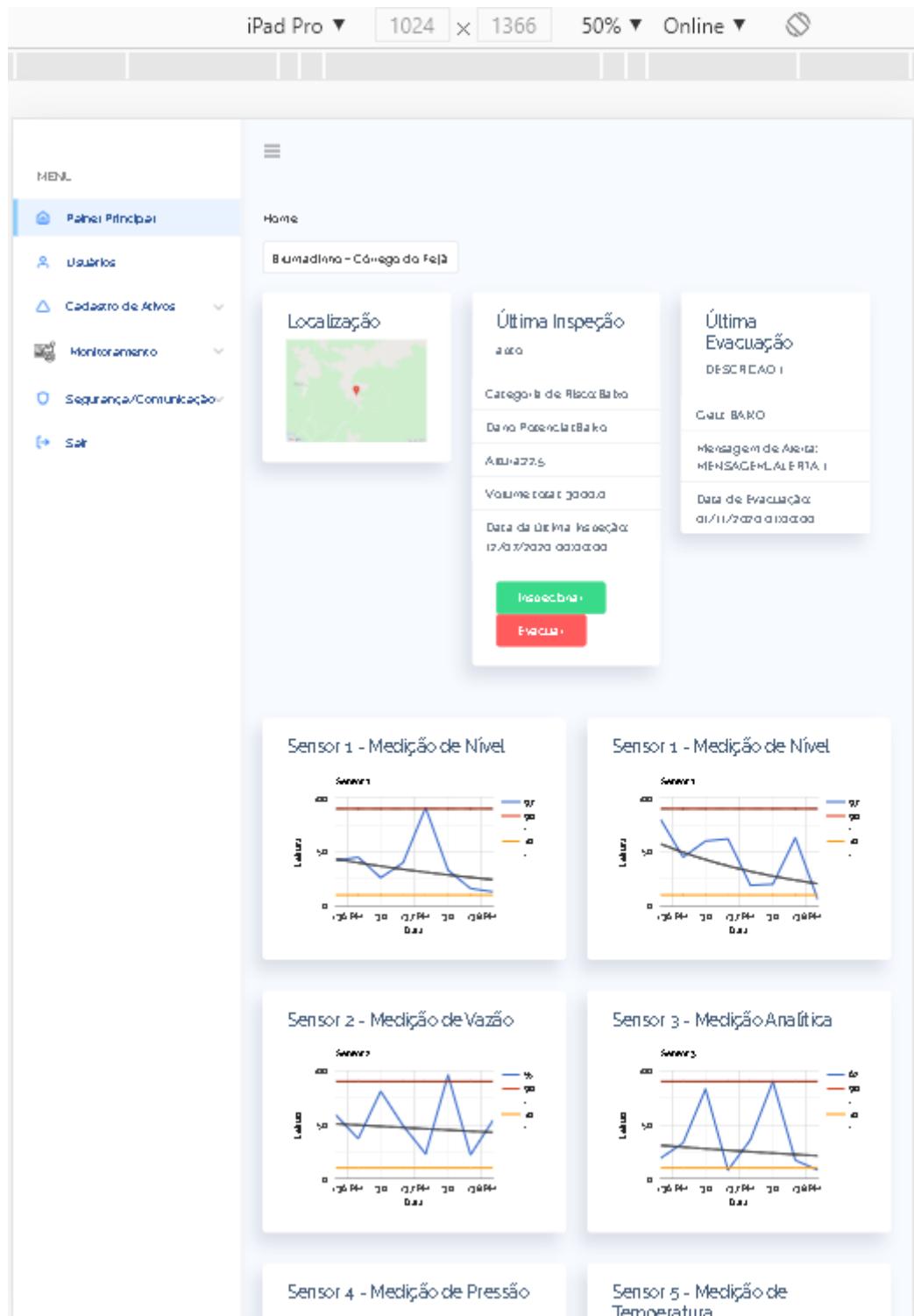


Figura 22 – Resolução 1024x1366 Ipad Pro.

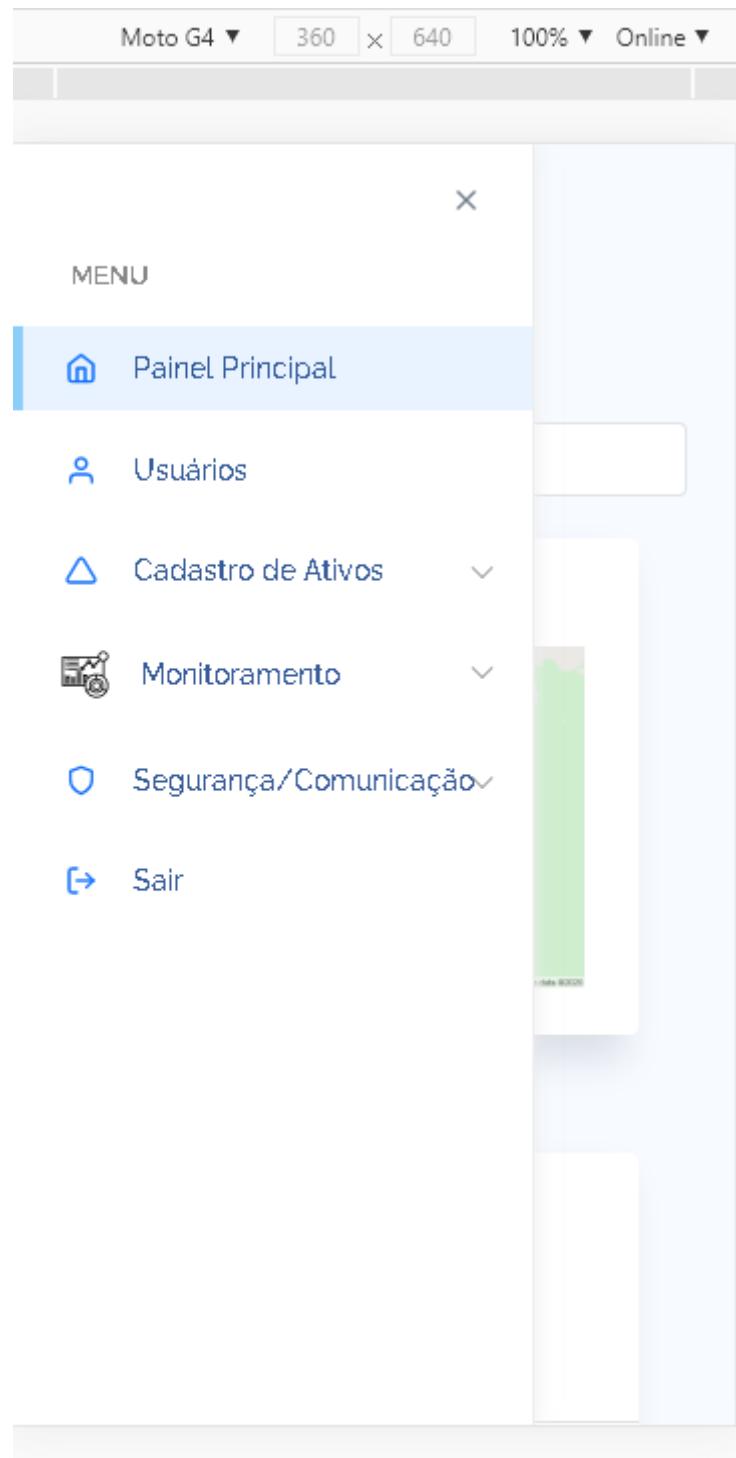


Figura 23 – Resolução 360x640 Moto G4.

Cenário 3:

Atributo de Qualidade:	Desempenho
Requisito de Qualidade:	O sistema deve possuir bom desempenho

Preocupação:	O sistema deve possuir um desempenho que não ultrapasse o limite estabelecido
Cenário(s):	Cenário 3
Ambiente:	Produção, carga normal
Estímulo:	Usuário cadastra um novo ativo
Mecanismo:	Camada frontend simples e objetiva e um backend otimizado em um servidor clusterizado.
Medida de Resposta:	A página possui um retorno menor que 5 segundos após a ação do usuário
Considerações sobre a arquitetura:	
Riscos	Pode ocorrer sobrecarga do servidor devido acessos simultâneos acima do normal ocasionando lentidão.
Pontos de Sensibilidade:	Servidor de aplicação
Tradeoff:	Não existe

Evidências do Cenário 3:

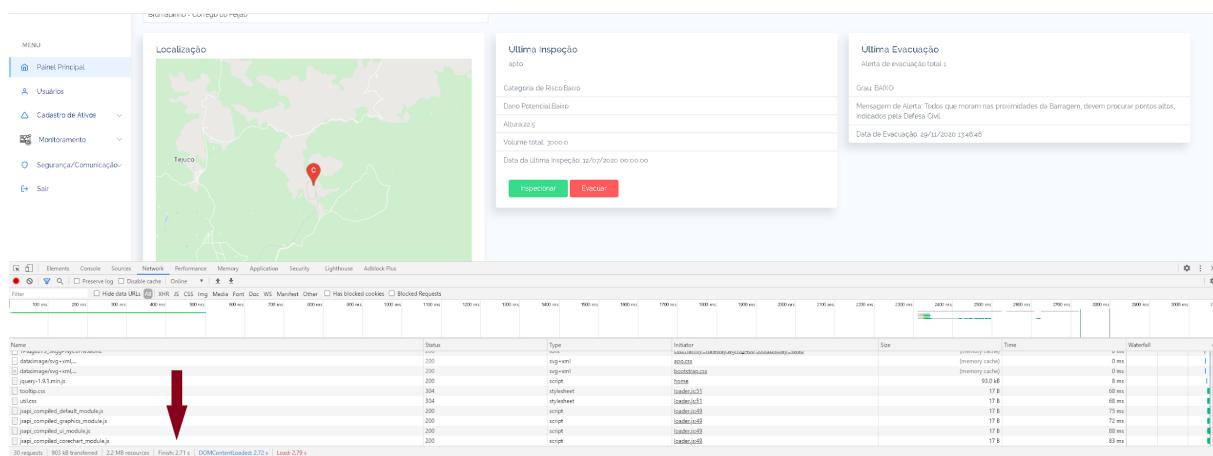


Figura 24– Evidência do retorno menor que 5 segundos após a ação do usuário.

Cenário 4:

Atributo de Qualidade:	Interoperabilidade
Requisito de Qualidade:	Envio de email em caso de notificação de incidente
Preocupação:	<p>Se os dados recebidos de algum sensor ultrapassar o mínimo ou o máximo estabelecidos, ou ainda se um engenheiro sinalizar a necessidade de evacuação o sistema deve enviar o email de alerta corretamente para os endereços pré-definidos.</p>
Cenário(s):	Cenário 4
Ambiente:	
Produção, carga normal	
Estímulo:	<p>O módulo de monitoramento de barragens recebe os dados dos sensores, caso algum destes dados tenham ultrapassado os limites (mínimos ou máximos) estabelecidos o módulo de segurança e comunicação é acionado. Existe também a possibilidade de um engenheiro, após identificar algum problema, solicitar a evacuação independente de leitura dos sensores. A integração com o módulo de segurança e comunicação funciona da mesma forma anteriormente dita.</p> <p>Então o módulo de segurança e comunicação ao receber a notificação de um alerta envia um email de acordo com o que foi configurado no plano de ação.</p>
Mecanismo:	O email é enviado corretamente para o endereço pré-definido
Medida de Resposta:	
Considerações sobre a arquitetura:	
Riscos	Falha no servidor de email ou atrasos devido a sobrecarga são um problema nas notificações de alertas.
Pontos de Sensibilidade:	Servidor de email
Tradeoff:	Não existe

Evidências do Cenário 4:

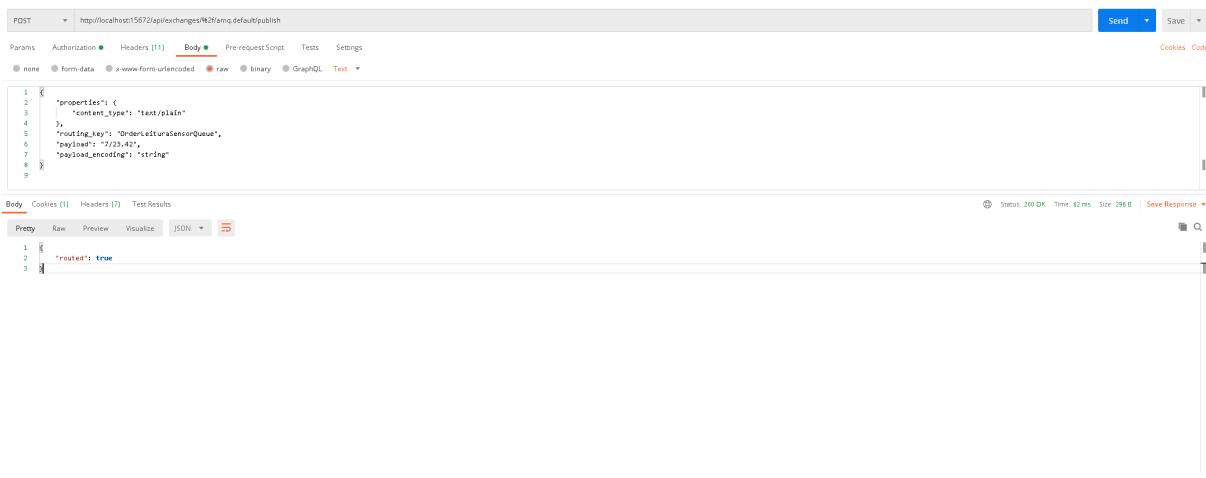


Figura 24 – Requisição do sensor, efetuado diretamente para o servidores de mensageria.



Figura 25 – Gráfico exibindo leituras do sensor com o Máximo e Mínimo para envio de e-mail.

MENU

- Panel Principal
- Usuarios
- Cadastro de Ativos
- Monitamento
- Segurança/Comunicação
 - Comunicação
 - Rane Ação
 - Pessoas
- Sair

Lista de Plano de Ação

1 resultados por página

#	Ativo	Pessoas	Descrição	Mensagem de Alerta	Grau de Risco	Data Inclusão	Data Inativação	Pesquisar	
1	Barragem de Rejetos 1	Vicente Benjamin Bryan da Rosa Raul Bruno da Luz Vicente Luiz Bernandes Luiza Sebastiana da Costa Carlos Caubá Monteiro	DESCRICAO 1	MENSAGEM_ALERTA 1	BAIXO	03/11/2020 03:00:00			
2	Barragem de Rejetos 2		DESCRICAO 2	MENSAGEM_ALERTA 2	MEDIO	03/11/2020 03:00:00			
3	Barragem de Rejetos 3		DESCRICAO 3	MENSAGEM_ALERTA 3	ALTO	03/11/2020 03:00:00			
4	Barragem de Rejetos 4		DESCRICAO 4	MENSAGEM_ALERTA 4	BAIXO	03/11/2020 03:00:00			
5	Barragem de Rejetos 5		DESCRICAO 5	MENSAGEM_ALERTA 5	MEDIO	03/11/2020 03:00:00			
6	Sensor 1	Vicente Benjamin Bryan da Rosa	DESCRICAO 6	MENSAGEM_ALERTA 6	BAIXO	03/11/2020 03:00:00			
7	Sensor 2	Raul Bruno da Luz	DESCRICAO 7	MENSAGEM_ALERTA 7	MEDIO	03/11/2020 03:00:00			
8	Sensor 3	Vicente Luiz Bernandes	DESCRICAO 8	MENSAGEM_ALERTA 8	ALTO	03/11/2020 03:00:00			
9	Sensor 4	Luiza Sebastiana da Costa	DESCRICAO 10	MENSAGEM_ALERTA 10	BAIXO	03/11/2020 03:00:00			

Figura 26 – Plano de ação cadastrado, caso sensor 1 ultrapasse o mínimo ou o máximo estipulados para o sensor 1.

MENU

- Panel Principal
- Usuarios
- Cadastro de Ativos
- Monitamento
 - Barragens
 - Sensores
 - Tipo Sensor
- Segurança/Comunicação
- Sair

Lista de Tipo de Sensor

1 resultados por página

#	Nome	Lerida Mínima	Lerida Máxima	Data Inclusão	Data Inativação	Pesquisar	
1	Medição de Nível	90.0	10.0	03/11/2020 03:00:00			
2	Medição de Vazão	90.0	10.0	03/11/2020 03:00:00			
3	Medição Analítica	90.0	10.0	03/11/2020 03:00:00			
4	Medição de Pressão	90.0	10.0	03/11/2020 03:00:00			
5	Medição de Temperatura	90.0	10.0	03/11/2020 03:00:00			
6	Deteção de Po	90.0	10.0	03/11/2020 03:00:00			

Figura 27 – Tipo de sensor, com o mínimo e máximo cadastrados.

Gmail

Caixa de entrada 1.250

Principal

Social

Promoções

Alerta de Sensor 9 - Emissão de alerta de sensor 9

Alerta de Sensor 10 - Emissão de alerta de sensor 10

Alerta de Sensor 8 - Emissão de alerta de sensor 8

Alerta de Sensor 6 - Emissão de alerta de sensor 6

Alerta de Sensor 7 - Emissão de alerta de sensor 7

Alerta de Sensor 9 - Emissão de alerta de sensor 9

Alerta de Sensor 8 - Emissão de alerta de sensor 8

Alerta de Sensor 10 - Emissão de alerta de sensor 10

Alerta de Sensor 7 - Emissão de alerta de sensor 7

Alerta de Sensor 6 - Emissão de alerta de sensor 6

DESCRICAO 6 - MENSAGEM_ALERTA 6

DESCRICAO 8 - MENSAGEM_ALERTA 8

DESCRICAO 10 - MENSAGEM_ALERTA 10

DESCRICAO 8 - MENSAGEM_ALERTA 8

DESCRICAO 7 - MENSAGEM_ALERTA 7

DESCRICAO 10 - MENSAGEM_ALERTA 10

DESCRICAO 7 - MENSAGEM_ALERTA 7

DESCRICAO 10 - MENSAGEM_ALERTA 10

DESCRICAO 10 - MENSAGEM_ALERTA 10

Figura 28 – E-mails enviados para os usuário cadastrados no plano de ação do sensor.

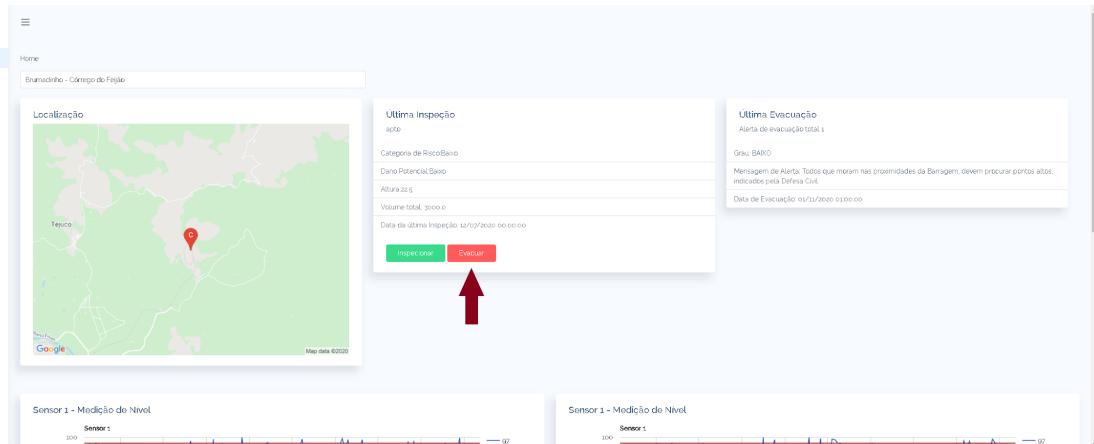


Figura 29 – Emissão de evacuação da barragem escolhida na tela principal.

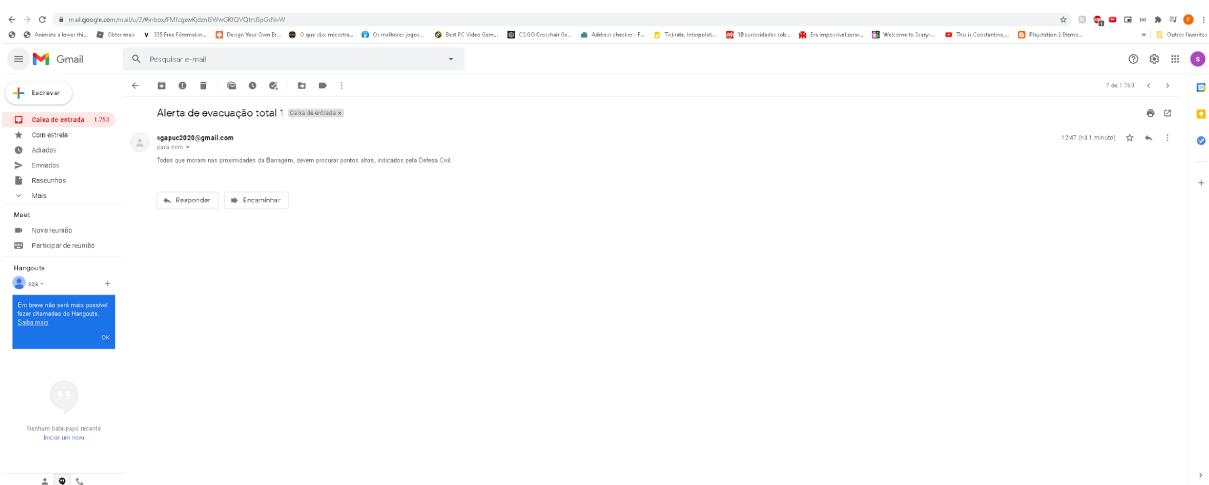


Figura 30 – E-mails enviados para os usuário cadastrados no plano de ação da Barragem.

6.4. Resultado

A validação arquitetural tem como objetivo a análise dos atributos de qualidade. Através desta prova de conceito foi constatado que necessidades principais do projeto fossem atendidas, com margem para futuras melhorias.

Nesta avaliação, os seguintes requisitos foram considerados:

Requisitos não funcionais	Testado	Homologado
O sistema deve ter alta segurança de seus módulos,	SIM	SIM

disponibilizando o acesso somente aos usuários autorizados.		
O sistema deve possuir o layout responsivo, com a finalidade de ter o acesso possível via desktop ou dispositivos móveis	SIM	SIM
O sistema deve possuir respostas rápidas as ações do usuário.	SIM	SIM
O sistema enviar email de notificação de alerta	SIM	SIM

O frontend foi pensando primeiramente na usabilidade e na adaptabilidade para diversos tipos de dispositivos. Isto significa que, utilizando uma versão moderna de navegador, o usuário consegue acessar o sistema através de diferentes tipos de aparelhos no mercado: desktops, notebooks, celulares e tablet de diferentes sistemas operacionais. A solução ideal seria a criação de aplicativos nativos para os dois principais SOs do mercado (Android e iOS) além dos da aplicação web, melhorando consideravelmente a usabilidade e possibilidades de funcionalidades que uma aplicação nativa pode oferecer. Entretanto a criação destes aplicativos aumentaria consideravelmente o custo do projeto. Em um primeiro momento a solução atual atende as necessidades estes usuários de dispositivos móveis, porém é uma possibilidade futura de expansão.

Para cada módulo do sistema foi criado um micro serviço, cada um contendo sua própria base de dados independente. Esta abordagem apresentou uma perda de comunicação e dependência direta de serviços que acabam por consumir uma parte muito substancial da memória disponível por máquina, o que demonstrou ser um ponto da arquitetura que pode ser melhorado no futuro. A separação do serviço é o lado positivo da independência ao banco de dados, podendo ser repensados futuramente para permitir que pequenas partes do sistema possam ser escaladas separadamente.

As imagens Docker são construídas através de arquivos de definição previamente configurados, podendo, após sua aplicação ser transformada numa imagem Docker, e são

instanciadas como container, em qualquer ambiente(host) que for necessário. Há maior garantia de que não haverá imprevistos ou erros durante o deploy. Gastando-se menos tempo para a configuração dos ambientes e reduzindo o tempo com análise e identificação das diferenças existentes entre sistemas. Em alguns momentos o uso do Docker apresentou muito esforço em suas configurações iniciais. Nesses casos, normalmente, o problema está mais em como a aplicação trabalha do que na configuração do Docker.

Há cada vez mais soluções que facilitam o deploy de forma simples e automatizada. Utilizando Trigger do GitHub e CodePipeline da AWS, consegue-se obter muitos benefícios no processo de build e deploy da aplicação, permitindo que correções sejam publicadas mais rapidamente, e que cada serviço seja implantado de maneira independentemente. Todos os testes unitários e testes de integração são executados no build, para garantir a integridade e estabilidade do que foi desenvolvido, ajudando a validar a implantação e permitindo a implantação somente após a validação de todos os testes.

7. Conclusão

O projeto como um todo, apresentou uma arquitetura robusta para um sistema de gestão ambiental eficiente. Entende-se que a arquitetura escolhida, foi plenamente capaz de satisfazer os requisitos principais da aplicação. As limitações identificadas não causaram impedimento na implementação, mas podem servir como ponto de atenção para próximas versões, possibilitando melhorias futuras, e que poderão ser adicionadas ao projeto sem grandes dificuldades.

REFERÊNCIAS

Agência Nacional de Mineração (ANM)

<<https://www.gov.br/anm/pt-br>>, 2020.

Spring Boot. Criando micro serviços com o Spring Boot.

<<https://www.infoq.com/br/articles...>>, 2020.

Docker. Docker Documentation. <<https://docs.docker.com/>>, 2020.

Docker Compose. O que é? Para que serve? O que come?. <

<<https://imasters.com.br/banco-de-dados...>>, 2020.

Modelo Componentes. Modelo de Componentes. <<https://dtic.tjpr.jus.br/wiki?...>>,

2020.

Modelagem de Sistemas. Modelagem de Sistemas. <

<<http://estacio.webaula.com.br/BiBlioTECA...>>, 2020.

Modelagem de Sistemas. Modelagem de Sistemas. <

<<http://estacio.webaula.com.br/BiBlioTECA...>>, 2020.

APÊNDICES

URL do repositório (Github) base para garantir a autenticidade dos trabalhos.

Link: <https://github.com/felipeBalbino/SCA-puc-minas>

URL do vídeo mostrando apresentação da POC.

Link: http://fjbalbino.com/sga_video/

URL do sistema da POC hospedado em nuvem..

Link: http://fjbalbino.com/sga_sistema/

Dados de acesso:

- Administrador

Usuário: administrador

Senha: qwe123

- Engenheiro

Usuário :engenheiro

Senha: qwe123

- Consultor

Usuário: consultor

Senha: qwe123